Università degli Studi di Genova

Scuola di scienze Matematiche, Fisiche e Naturali

Corso di Laurea Magistrale in
Matematica Applicata

Anno accademico 2023/2024

Tesi di Laurea

Introduzione a FALCON

Candidata
Janet Geraci

Relatore
Alessio Caminata

Indice

Introduzione

I crittosistemi a chiave pubblica attualmente in uso (RSA, ECC, Diffie-Hellman, El Gamal...) non sono più considerati sicuri a causa dell'avvento del computer quantistico: i problemi alla base di essi sono risolvibili in tempi impraticabili da un computer classico, mentre per un computer quantistico sono stati elaborati algoritmi che rendono tali problemi veloci da risolvere; da qui il problema di cercare nuovi crittosistemi resistenti anche all'attacco di un tale calcolatore.

Il NIST (National Institute of Technology) è un'agenzia governativa statunitense che gestisce le tecnologie di diversi ambiti, inclusa la Cybersecurity; essa promulga standard da seguire ad esempio per la sicurezza crittografica.

Nel 2016 indisse una sfida tra i crittografi di tutto il mondo per poter standardizzare nuove costruzioni crittografiche, resistenti ai computer quantistici.

FALCON figura tra le varie proposte per schemi di firma digitali, ed è uno dei tre schemi ad aver passato tutte le selezioni, finendo tra i vincitori della sfida. È considerato dunque una possibile concreta alternativa a quanto in uso oggi.

FALCON è un acronimo che deriva da

Fast Fourier lattice-based compact signatures over NTRU

ossia firme compatte basate su reticoli NTRU con Fast Fourier. Si tratta di uno schema di firma su una particolare classe di reticoli che utilizza una nuova tecnica di campionamento ispirata alla trasformata di Fourier veloce discreta.

Fu ideato da Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Prest, Thomas Ricosset, Gregor Seiler, William Whyte, Zhenfei Zhang e fu sottoposto al giudizio del NIST il 30 novembre del 2017. Si può consultare la documentazione relativa al sito ufficiale FALCON.

Il pensiero alla base di questo schema è il seguente: quando si dovrà passare dagli schemi di firma classici a quelli cosiddetti post-quantum, oltre alla sicurezza sarà importante anche la praticità: nonostante molte nuove costruzioni siano semplici, rendendo l'implementazione relativa veloce, esse necessitano molto spazio per immagazzinare le chiavi e/o le firme. Quindi l'obiettivo è minimizzare contemporaneamente la lunghezza in bit della chiave pubblica e quella delle firme. Seguendo questo principio, si sceglie di instaurare questa costruzione non su reticoli generici (mettendo al primo posto la sicurezza), ma su reticoli strutturati,

che permettono di velocizzare le operazioni e ridurre notevolmente lo spazio per immagazzinare la chiave pubblica.

L'impalcatura di questo schema è quella designata da Gentry, Peikert e Vaikuntanathan allo STOC 2008, a cui si aggiunge però una nuova tecnica di campionamento da reticoli, ispirata alla tecnica veloce di calcolo della trasformata di Fourier discreta.

L'obiettivo di questa tesi è descrivere in modo più accessibile FALCON e i lavori precedenti che hanno portato ad esso. Nel capitolo 1 si forniscono le prime basi sui reticoli, nel capitolo 2 e 3 si illustrano rispettivamente i crittositemi e gli schemi di firma basati sui reticoli che furono il punto di partenza per arrivare a questo risultato. Il capitolo 4 tratta il GPV framework, ossia l'ossatura dello schema. Il capitolo 5 presenta la novità più importante di FALCON, che favorisce la sua compattezza, ossia un algoritmo di campionamento da reticoli allo stesso più sicuro e più efficiente. Infine, il capitolo 6 descrive lo schema FALCON nella sua interezza.

Notazioni

$\mathbb{N}$ indica l'insieme dei numeri naturali (incluso lo 0), $\mathbb{Z}$ indica l'insieme dei numeri interi,

$\mathbb{Q}$ indica i numeri razionali, $\mathbb{R}$ i reali e $\mathbb{C}$ i complessi.

Si indicano con lettere maiuscole le matrici (ad esempio $B \in \mathbb{R}^{n \times m}$), con lettere minuscole gli scalari ($a \in \mathbb{Q}$) e con lettere minuscole sottolineate i vettori ($\underline{v} \in \mathbb{Z}^n$).

Dato $x \in \mathbb{C}$, si indica con $\overline{x}$ la sua coniugazione complessa.

Con $\delta_{i,j}$ si indica il **delta di Kronecker**:

$$\delta_{i,j} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \tag{1}$$

Notazione di Landau Siano $f, g : \mathbb{N} \to \mathbb{R}$.

Si scrive

$$f(n) = O(g(n)) \tag{2}$$

per $n \to \infty$ se $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \ell \in \mathbb{R}$.

Si scrive

$$f(n) = o(g(n)) \tag{3}$$

per $n \to \infty$ se $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$.

Si scrive

$$f(n) = \omega(g(n)) \tag{4}$$

se $\forall\ C > 0\ \exists\ N : \forall n > N\ |f(n)| \geq C|g(n)|$.

Si scrive

$$f(n) = \Theta(g(n)) \tag{5}$$

per $n \to \infty$ se $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \ell \in \mathbb{R} \setminus \{0\}$.

Si indica

$$f(n) = \text{negl}(n) \tag{6}$$

se $f(n) = o(n^{-c})$ per ogni costante fissata $c > 0$, e si dice f è trascurabile.

Sia D una certa distribuzione di probabilità. Si scrive

$$x \sim D \tag{7}$$

se x è campionato da tale distribuzione.

$\approx$ si sostituisce a $=$ quando l'uguaglianza vale in modo approssimato.

Distanza statistica Date due distribuzioni X e Y su un dominio numerabile D, la distanza statistica tra le due è la quantità

$$\frac{1}{2}\sum_{d\in D}|X(d)-Y(d)| \tag{8}$$

Dati due interi n, m si indica con $\gcd(n, m)$ il massimo comun divisore tra n e m.

Dato x un elemento di $\mathbb{R}$, si indica con $\lfloor x \rceil$ l'intero più vicino a x, coerentemente con le notazioni per parte intera inferiore di x ($\lfloor x \rfloor$) e parte intera superiore di x ($\lceil x \rceil$). Nel caso in cui $x - \lfloor x \rfloor$ sia uguale a $\frac{1}{2}$, si sceglie arbitrariamente $\lfloor x \rceil = \lceil x \rceil$.

Questa notazione si estende anche a $\underline{v} \in \mathbb{R}^n$ componente per componente.

Capitolo 1

Teoria di base dei reticoli

1.1 Reticoli

Un sottogruppo $\mathcal{H}$ di $\mathbb{R}^n$ si dice **discreto** se la topologia indotta su $\mathcal{H}$ dalla topologia euclidea di $\mathbb{R}^n$ è la topologia discreta. Tale condizione è equivalente al fatto che l'intersezione di $\mathcal{H}$ con un qualsiasi compatto di $\mathbb{R}^n$ abbia cardinalità finita.

Se $\mathcal{H}$ è un sottogruppo discreto di $\mathbb{R}^n$, allora esistono r vettori linearmente indipendenti su $\mathbb{R}$ che generano $\mathcal{H}$ come $\mathbb{Z}$-modulo (quindi $r \leq n$). Per vedere una dimostrazione di questi fatti si rimanda a [23] oppure a [22].

Definizione 1.1.1. *Si dice* ***reticolo*** *di dimensione n un sottogruppo discreto di $\mathbb{R}^n$ di rango massimo.*

Alcuni autori definiscono i reticoli senza imporre la condizione sul rango; ciò si considera non necessario nella seguente trattazione, perché si considereranno solo reticoli di rango pieno.

Un reticolo Λ su $\mathbb{R}^n$ è dunque generato come $\mathbb{Z}$-modulo da una base di $\mathbb{R}^n$. Data una tale base $\mathcal{B} = \{\underline{v}_1, \ldots, \underline{v}_n\}$, allora ogni vettore di Λ si può esprimere come combinazione lineare dei vettori di $\mathcal{B}$ a coefficienti in $\mathbb{Z}$:

$$\Lambda = \left\{ \sum_{i=1}^{n} \alpha_i \underline{v}_i : \alpha_i \in \mathbb{Z} \right\}, \tag{1.1}$$

dove un qualsiasi insieme di generatori linearmente indipendenti per Λ si dice **base** del reticolo. Una **basis matrix** B per Λ è una matrice avente come righe gli elementi di una base di Λ. Allora Λ si può anche scrivere come

$$\Lambda = \{\underline{\alpha} B : \underline{\alpha} \in \mathbb{Z}^n\}. \tag{1.2}$$

Si considereranno solo **reticoli interi**, cioè reticoli in cui tutti i vettori hanno entrate intere; è sufficiente che i vettori della base abbiano entrate intere per assicurare ciò.

Dati $\{\underline{w}_1, \ldots, \underline{w}_n\}$ in Λ, ognuno di essi si può scrivere come

$$\underline{w}_i = \sum_{i=1}^{n} \alpha_{i,j}\underline{v}_j \text{ con } \alpha_{i,j} \in \mathbb{Z} \text{ per ogni } i, j \tag{1.3}$$

Chiamando B la matrice con righe gli elementi di $\mathcal{B}$, B' la matrice con righe $\{\underline{w}_1, \ldots, \underline{w}_n\}$, e $A = (\alpha_{i,j})_{i,j} \in M_n(\mathbb{Z})$ si ha che

$$B' = AB \tag{1.4}$$

Se le righe di B' sono linearmente indipendenti, sono anch'esse una base di $\mathbb{R}^n$, e quindi

$$B = A'B' \text{ con } A' \in M_n(\mathbb{R}) \tag{1.5}$$

Le righe di B' generano anche il reticolo solo se $A' \in M_n(\mathbb{Z})$, si considera questo caso. A e A' sono le due matrici di cambiamento di base, perciò sono l'una l'inversa dell'altra. Sfruttando ciò, si ottiene che:

$$1 = \det(I) = \det(AA') = \det(A)\det(A') \tag{1.6}$$

Dato che entrambe le matrici hanno entrate intere, esse hanno come determinante un elemento di $\mathbb{Z}$ invertibile; l'unica possibilità è che

$$\det(A) = \det(A') = \pm 1. \tag{1.7}$$

Si è appena mostrato che ogni matrice di cambio di base di un reticolo ha entrate in $\mathbb{Z}$ e determinante ± 1.

Definizione 1.1.2. *Sia $\Lambda \subseteq \mathbb{R}^n$ un reticolo con base $\mathcal{B} = \{\underline{v}_1, \ldots, \underline{v}_n\}$. Si dice* ***dominio fondamentale*** *per Λ rispetto a $\mathcal{B}$*

$$\mathscr{F}_{\mathcal{B}} = \left\{ \underline{x} \in \mathbb{R}^n : \underline{x} = \sum_{i=1}^{n} t_i\underline{v}_i,\ 0 \le t_i < 1 \right\} \tag{1.8}$$

Lemma 1.1.3. *Dato un dominio fondamentale $\mathscr{F}_{\mathcal{B}}$ per Λ*

$$\mu\left(\mathscr{F}_{\mathcal{B}}\right) = |\det(B)| \tag{1.9}$$

dove con B si indica la matrice con righe gli elementi della base $\mathcal{B}$ e con μ la misura di Lebesgue su $\mathbb{R}^n$.

Dimostrazione.

$$\mu\left(\mathscr{F}_{\mathcal{B}}\right) = \int_{\mathscr{F}_{\mathcal{B}}} dx_1 \ldots dx_n = \int_{[0,1)^n} |\det(B)|\ dt_1 \ldots dt_n = |\det(B)| \tag{1.10}$$

effettuando il cambio di variabili

$$(x_1, \ldots, x_n) = t_1v_1 + \cdots + t_nv_n = (t_1, \ldots, t_n)B$$

con matrice Jacobiana B e sfruttando che la misura dell'insieme $[0,1)^n$ è 1. □

Sfruttando quanto appena visto

$$\mu\left(\mathscr{F}_{\mathcal{B}'}\right) = |\det(B')| \overset{(1.4)}{=} |\det(AB)| = |\det(A)||\det(B)| \overset{(1.7)}{=} |\det(B)| = \mu\left(\mathscr{F}_{\mathcal{B}}\right)$$

Perciò il volume di un dominio fondamentale per Λ non dipende dalla scelta della base e si può dare la seguente definizione.

Definizione 1.1.4. *Dato un reticolo Λ in $\mathbb{R}^n$, si dice* ***determinante*** *di Λ il volume di uno qualsiasi dei suoi domini fondamentali. Esso si indica con* $\det(\Lambda)$.

Teorema 1.1.5. *Sia Λ un reticolo di dimensione n e sia $\mathscr{F}$ un dominio fondamentale per Λ. Ogni elemento $\underline{w}$ di $\mathbb{R}^n$ si scrive in maniera unica come*

$$\underline{w} = \underline{v} + \underline{t}$$

con $\underline{v}$ un elemento di Λ e $\underline{t}$ un elemento di $\mathscr{F}$. In altre parole, $\underline{w}$ appartiene a $\mathscr{F} + \underline{v}$ (dominio fondamentale traslato di $\underline{v}$).

Dimostrazione. Dato che una qualsiasi base $\mathcal{B}$ di Λ è anche base di $\mathbb{R}^n$, $\underline{w}$ si può scrivere come combinazione lineare dei suoi elementi a coefficienti in $\mathbb{R}$:

$$\underline{w} = a_1\underline{v}_1 + \cdots + a_n\underline{v}_n \qquad a_i \in \mathbb{R} \tag{1.11}$$

Ogni coefficiente a_i si può vedere come la somma della sua parte intera inferiore e della sua parte decimale, ossia $a_i = \lfloor a_i \rfloor + (a_i - \lfloor a_i \rfloor) = \alpha_i + s_i$ con $\alpha_i \in \mathbb{Z}$ e $0 \leq s_i < 1$.

$$w = \underbrace{\alpha_1\underline{v}_1 + \cdots + \alpha_n\underline{v}_n}_{\in \Lambda} + \underbrace{s_1\underline{v}_1 + \cdots + s_n\underline{v}_n}_{\in \mathscr{F}} \tag{1.12}$$

da cui la tesi. Per l'unicità, si supponga che

$$\underline{w} = \underline{v} + \underline{t} = \hat{\underline{v}} + \hat{\underline{t}} \tag{1.13}$$

con $\underline{v}, \hat{\underline{v}} \in \Lambda$ e $\underline{t}, \hat{\underline{t}} \in \mathscr{F}$. Allora (1.13) è equivalente a

$$(\alpha_1 + s_1)\underline{v}_1 + \cdots + (\alpha_n + s_n)\underline{v}_n = (\hat{\alpha_1} + \hat{s_1})\underline{v}_1 + \cdots + (\hat{\alpha_n} + \hat{s_n})\underline{v}_n \tag{1.14}$$

Poiché i vettori della base sono linearmente indipendenti, questa uguaglianza si traduce nelle uguaglianze

$$\alpha_i + s_i = \hat{\alpha}_i + \hat{s}_i \qquad \text{per ogni } i \in \{1, \ldots, n\} \tag{1.15}$$

o equivalentemente

$$s_i - \hat{s}_i = \hat{\alpha}_i - \alpha_i \in \mathbb{Z} \qquad \text{per ogni } i$$

Dato che $0 \leq s_i, \hat{s}_i < 1$, affinché la loro differenza sia intera è necessario che $s_i - \hat{s}_i = 0$, il che implica che $\alpha_i - \hat{\alpha}_i = 0$, il che conclude la prova. □

Il precedente teorema afferma quindi che è possibile tassellare tutto lo spazio $\mathbb{R}^n$ con domini fondamentali.

1.1.1 Ortogonalizzazione di Gram-Schmidt

Notazione 1.1.6. Si indica con $\langle \cdot, \cdot \rangle_2$ il prodotto scalare di $\mathbb{R}^n$ e con $\| \cdot \|_2$ la norma indotta da tale prodotto scalare; ci si riferisce ad essa come lunghezza di un vettore. Si ricorda che, per ogni vettore $\underline{v}$ appartenente a $\mathbb{R}^n$, $\langle \underline{v}, \underline{v} \rangle_2 = \|\underline{v}\|_2^2$.

Inoltre, due vettori $\underline{v}$ e $\underline{w}$ si dicono ortogonali o perpendicolari tra loro se $\langle \underline{v}, \underline{w} \rangle_2 = 0$.

L'ortogonalizzazione di Gram-Schmidt è un algoritmo che, a partire da un insieme di vettori linearmente indipendenti $\{\underline{v}_1, \ldots, \underline{v}_n\}$ in $\mathbb{R}^m$, restituisce $\{\underline{v}_1^*, \ldots, \underline{v}_n^*\}$, vettori a due a due ortogonali, dove per ogni $i = 1, \ldots, n$

$$\underline{v}_i^* = \underline{v}_i - \sum_{j=1}^{i-1} \mu_{i,j} \underline{v}_j^* = \underline{v}_i - \sum_{j=1}^{i-1} \frac{\langle \underline{v}_i, \underline{v}_j^* \rangle_2}{\|\underline{v}_j^*\|_2^2} v_j^* \tag{1.16}$$

Algorithm 1 Algoritmo di ortogonalizzazione di Gram-Schmidt

Input: $\{\underline{v}_1, \ldots, \underline{v}_n\}$ linearmente indipendenti
Output: $\{\underline{v}_1^*, \ldots, \underline{v}_n^*\}$ ortogonali

```
v_1* ← v_1
for i=2, ..., n do
    v ← v_i
    for j=2, ..., i-1 do
        μ_{i,j} ← ⟨v_i, v_j*⟩_2 / ||v_j*||_2^2
        v ← v − μ_{i,j} v_j*
    end for
    v_i* ← v
end for
return {v_1*, ..., v_n*}
```

Lemma 1.1.7. *Sia $1 \leq j < k \leq n$, allora $\langle \underline{v}_j^*, \underline{v}_k^* \rangle_2 = 0$, ossia i vettori restituiti dall'algoritmo di Gram-Schmidt sono a due a due perpendicolari. Inoltre,*

$$\text{span}\{\underline{v}_1, \ldots, \underline{v}_i\} = \text{span}\{\underline{v}_1^*, \ldots, \underline{v}_i^*\} \tag{1.17}$$

per ogni $i \in \{1, \ldots, n\}$. In altre parole, lo spazio generato dai primi i vettori originali è uguale allo spazio generato dai primi i vettori calcolati con l'algoritmo per ogni i.

Dimostrazione. Si dimostra la prima parte dell'enunciato per induzione su k.

k=2 Se $k = 2$, allora $j = 1$. Di conseguenza si ha:

$$\underline{v}_1^* = \underline{v}_1$$
$$\underline{v}_2^* = \underline{v}_2 - \frac{\langle \underline{v}_2, \underline{v}_1^* \rangle_2}{\|\underline{v}_1^*\|^2} \underline{v}_1^* = \underline{v}_2 - \frac{\langle \underline{v}_2, \underline{v}_1 \rangle_2}{\|\underline{v}_1\|^2} \underline{v}_1$$

Sfruttando le identità precedenti si calcola il prodotto scalare tra $\underline{v}_1^*$ e $\underline{v}_2^*$:

$$\begin{aligned}\langle \underline{v}_1^*, \underline{v}_2^* \rangle_2 &= \langle \underline{v}_1, \underline{v}_2 - \frac{\langle \underline{v}_2, \underline{v}_1 \rangle_2}{\|\underline{v}_1\|_2^2} \underline{v}_1 \rangle_2 = \\ &= \langle \underline{v}_1, \underline{v}_2 \rangle_2 - \frac{\langle \underline{v}_2, \underline{v}_1 \rangle_2}{\|\underline{v}_1\|_2^2} \underbrace{\langle \underline{v}_1, \underline{v}_1 \rangle_2}_{=\|\underline{v}_1\|_2^2} = \\ &= \langle \underline{v}_1, \underline{v}_2 \rangle_2 - \langle \underline{v}_1, \underline{v}_2 \rangle_2 = 0\end{aligned}$$

k>2 Si suppone la tesi vera fino a $k = n-1$ e si dimostra il caso $k = n$ per ogni $j < n$.

$$\underline{v}_n^* = \underline{v}_n - \sum_{l=1}^{n-1} \mu_{n,l} \underline{v}_l^* = \underline{v}_n - \sum_{l=1}^{n-1} \frac{\langle \underline{v}_n, \underline{v}_l^* \rangle_2}{\|\underline{v}_l^*\|_2^2} \underline{v}_l^*$$

$$\begin{aligned}\langle \underline{v}_j^*, \underline{v}_n^* \rangle &= \langle \underline{v}_j^*, \underline{v}_n \rangle_2 - \sum_{l=1}^{n-1} \frac{\langle \underline{v}_n, \underline{v}_l^* \rangle_2}{\|\underline{v}_l^*\|_2^2} \langle \underline{v}_j^*, \underline{v}_l^* \rangle_2 = \\ &= \langle \underline{v}_j^*, \underline{v}_n \rangle_2 - \sum_{l=1, l \neq j}^{n-1} \frac{\langle \underline{v}_n, \underline{v}_l^* \rangle_2}{\|\underline{v}_l^*\|^2} \underbrace{\langle \underline{v}_j^*, \underline{v}_l^* \rangle_2}_{=0} - \frac{\langle \underline{v}_n, \underline{v}_j^* \rangle_2}{\|\underline{v}_j^*\|^2} \underbrace{\langle \underline{v}_j^*, \underline{v}_j^* \rangle_2}_{=\|\underline{v}_j^*\|_2^2} = \\ &= \langle \underline{v}_j^*, \underline{v}_n^* \rangle_2 - \langle \underline{v}_j^*, \underline{v}_n^* \rangle_2 = 0\end{aligned}$$

Per quanto riguarda la seconda parte dell'enunciato, si osserva innanzitutto che un contenimento è chiaro perché per ogni k

$$\underline{v}_k = \underline{v}_k^* + \sum_{j=1}^{k-1} \mu_{k,j} \underline{v}_j^*,$$

da cui

$$\{\underline{v}_1, \ldots, \underline{v}_i\} \subseteq \mathrm{span}\{\underline{v}_1^*, \ldots, \underline{v}_i^*\}$$

e a maggior ragione

$$\mathrm{span}\{\underline{v}_1, \ldots, \underline{v}_i\} \subseteq \mathrm{span}\{\underline{v}_1^*, \ldots, \underline{v}_i^*\}.$$

Per il viceversa si procede per induzione su i. Il caso $i = 1$ è banale perché $\underline{v}_1^* = \underline{v}_1$.

Il caso $i = 2$ sfrutta il caso precedente:

$$\underline{v}_2^* = \underline{v}_2 - \mu_{2,1}\underline{v}_1^* = \underline{v}_2 - \mu_{2,1}\underline{v}_1.$$

Il caso generale suppone la tesi vera fino a $i-1$ e sfrutta che

$$\underline{v}_i^* = \underline{v}_i - \sum_{j=1}^{i-1} \mu_{i,j}\underline{v}_j^*,$$

il che conclude la prova.

□

Una conseguenza del lemma precedente è che i vettori ottenuti dall'algoritmo sono più corti dei vettori iniziali, cioè

$$\|\underline{v}_i^*\| \leq \|\underline{v}_i\| \tag{1.18}$$

per ogni $i = 1, \ldots, n$. Infatti

$$\begin{aligned}
\|\underline{v}_i^*\|_2^2 &= \left\| \underline{v}_i - \sum_{j=1}^{i-1} \frac{\langle \underline{v}_i, \underline{v}_j^* \rangle_2}{\|\underline{v}_j^*\|_2^2} \underline{v}_j^* \right\|_2^2 = \\
&= \|\underline{v}_i\|_2^2 + \left\| \sum_{j=1}^{i-1} \frac{\langle \underline{v}_i, \underline{v}_j^* \rangle_2}{\|\underline{v}_j^*\|_2^2} \underline{v}_j^* \right\|_2^2 - 2 \left\langle \underline{v}_i, \sum_{j=1}^{i-1} \frac{\langle \underline{v}_i, \underline{v}_j^* \rangle_2}{\|\underline{v}_j^*\|_2^2} \underline{v}_j^* \right\rangle_2 \leq \\
&\leq \|\underline{v}_i\|_2^2 + \sum_{j=1}^{i-1} \left\| \frac{\langle \underline{v}_i, \underline{v}_j^* \rangle_2}{\|\underline{v}_j^*\|_2^2} \underline{v}_j^* \right\|_2^2 - 2 \sum_{j=1}^{i-1} \frac{\langle \underline{v}_i, \underline{v}_j^* \rangle_2}{\|\underline{v}_j^*\|_2^2} \langle \underline{v}_i, \underline{v}_j^* \rangle_2 = \\
&= \|\underline{v}_i\|^2 + \sum_{j=1}^{i-1} \frac{\langle \underline{v}_i, \underline{v}_j^* \rangle_2^2}{\|\underline{v}_j^*\|_2^4} \|\underline{v}_j^*\|_2^2 - 2 \sum_{j=1}^{i-1} \frac{\langle \underline{v}_i, \underline{v}_j^* \rangle_2^2}{\|\underline{v}_j^*\|_2^2} = \\
&= \|\underline{v}_i\|_2^2 - \sum_{j=1}^{i-1} \underbrace{\frac{\langle \underline{v}_i, \underline{v}_j^* \rangle_2^2}{\|\underline{v}_j^*\|_2^2}}_{\geq 0} \leq \\
&\leq \|\underline{v}_i\|_2^2
\end{aligned}$$

Osservazione 1.1.8. Visto che non è detto che i coefficienti $\mu_{i,j}$ siano interi, non è detto che un reticolo abbia una base ortogonale. Ci si accontenta considerando come buona base per un reticolo una tale che i suoi vettori siano quasi ortogonali e corti. Come mostra il lemma precedente, le due richieste sono tra loro collegate.

Osservazione 1.1.9. Data una matrice $B \in M_n(\mathbb{R})$ con righe $\{\underline{v}_1, \ldots, \underline{v}_n\}$, si definisce la **matrice di Gram** di B la matrice avente entrate $\langle \underline{v}_i, \underline{v}_j \rangle_2$ al variare di i e j in $\{1, \ldots, n\}$, cioè BB^T. Il determinante di tale matrice si può calcolare nel seguente modo:

$$\det((\langle \underline{v}_i, \underline{v}_j \rangle_2)_{i,j}) = \det(BB^T) = \det(B)\det(B^T) = \det(B)^2 \tag{1.19}$$

Lemma 1.1.10. *Sia Λ un reticolo di $\mathbb{R}^n$ con base $\mathcal{B} = \{\underline{v}_1, \ldots, \underline{v}_n\}$ e sia $\{\underline{v}_1^*, \ldots, \underline{v}_n^*\}$ l'output dell'algoritmo di Gram-Schmidt applicato a $\mathcal{B}$. Allora*

$$\det(\Lambda) = \prod_{i=1}^{n} \|\underline{v}_i^*\|_2 \tag{1.20}$$

Dimostrazione. Sia B^* la matrice avente come righe $\{\underline{v}_1^*, \ldots, \underline{v}_n^*\}$, ossia la matrice avente righe l'ortogonalizzazione di Gram-Schmidt di $\mathcal{B}$.

Allora

$$B^* = UB, \tag{1.21}$$

dove B è la basis matrix con righe gli elementi di $\mathcal{B}$ e U è una matrice triangolare inferiore avente entrate tutte uguali a 1 nella diagonale principale, il che implica che $\det(U) = 1$.

$$\begin{aligned} \det(\Lambda) &= |\det(B)| = && \text{per definizione,} \\ &= |\det(U)||\det(B)| = && \det(U) = 1, \\ &= |\det(UB)| = && \text{per Binet,} \\ &= |\det(B^*)| && \text{per (1.21).} \end{aligned}$$

Sfruttando (1.19)

$$|\det(B^*)| = \sqrt{\det(B^* B^{*^T})} = \sqrt{\det((\langle \underline{v}_i^*, \underline{v}_j^* \rangle_2)_{i,j})}.$$

Inoltre, per il lemma 1.1.7

$$\langle \underline{v}_i^*, \underline{v}_j^* \rangle_2 = \|\underline{v}_i^*\|_2^2 \delta_{i,j},$$

perciò

$$(\langle \underline{v}_i^*, \underline{v}_j^* \rangle_2)_{i,j}) = \mathrm{diag}(\|\underline{v}_1^*\|_2^2, \ldots, \|\underline{v}_n^*\|_2^2).$$

Quindi

$$\det(\Lambda) = \sqrt{\det(\mathrm{diag}(\|\underline{v}_1^*\|_2^2, \ldots, \|\underline{v}_n^*\|_2^2))} = \prod_{i=1}^{n} \|\underline{v}_i^*\|_2.$$

□

Da (1.20) unito a (1.18), si ottiene la cosiddetta **disuguaglianza di Hadamard**

$$\det(\Lambda) \leq \prod_{i=1}^{n} \|\underline{v}_i\|_2 \tag{1.22}$$

Vale l'uguaglianza se e solo se i vettori $\{\underline{v}_1, \ldots, \underline{v}_n\}$ sono a due a due ortogonali.

Definizione 1.1.11. *Dato Λ un reticolo di dimensione n con base $\mathcal{B} = \{\underline{v}_1, \ldots, \underline{v}_n\}$, si dice* ***Hadamard Ratio*** *della base $\mathcal{B}$*

$$\mathcal{H}(\mathcal{B}) = \left(\frac{\det(\Lambda)}{\|\underline{v}_1\|_2 \cdots \|\underline{v}_n\|_2} \right)^{1/n}$$

Si tratta di un indice della bontà di una base di Λ. Esso è maggiore di 0 e minore o uguale di 1, è uguale a 1 se e solo se la base è ortogonale. Più i vettori della base sono tra loro ortogonali, più tale valore si avvicina a 1. Una base viene considerata buona se $\mathcal{H}(\mathcal{B}) \geq 0.9$.

1.2 Problemi computazioni sui reticoli

Il motivo per il quale si ricerca una buona base per un reticolo Λ è che essa rende più semplice il trattamento di alcuni problemi computazionali basati sui reticoli, di cui fanno parte:

SVP (Shortest Vector Problem). *Dato un reticolo Λ di dimensione n, determinare un vettore non nullo più corto di Λ, ossia che risolva*

$$\min_{v \in \Lambda} \|\underline{v}\|_2$$

CVP (Closest Vector Problem). *Dato un reticolo Λ di dimensione n e un vettore $\underline{w}$ che non vi appartenga, determinare un vettore di Λ più vicino a $\underline{w}$, ossia che risolva*

$$\min_{\underline{v} \in \Lambda} \|\underline{v} - \underline{w}\|_2$$

apprSVP (Approximate Shortest Vector Problem). *Dato un reticolo Λ di dimensione n, determinare un vettore $\underline{v}_\psi$ di Λ tale che la sua lunghezza sia minore o uguale di $\psi(n)$ volte la lunghezza del vettore non nullo più corto di Λ, ossia che risolva*

$$\|\underline{v}_\psi\|_2 \leq \psi(n) \min_{\underline{v} \in \Lambda} \|\underline{v}\|_2$$

dove $\psi(n)$ è una funzione in n, ad esempio $3\sqrt{n}$ o $2^{n/2}$.

apprCVP (Approximate Closest Vector Problem). *Dato un reticolo Λ di dimensione n e un vettore $\underline{w}$ che non vi appartenga, determinare un vettore $\underline{v}_\psi$ di Λ tale che la sua distanza da $\underline{w}$ sia minore o uguale alla distanza di Λ da $\underline{w}$, ossia che risolva*

$$\|\underline{v}_\psi - \underline{w}\|_2 \leq \psi(n) \min_{\underline{v}\in\Lambda} \|\underline{v} - \underline{w}\|_2$$

dove $\psi(n)$ è una funzione in n.

Osservazione 1.2.1. Scegliendo $\underline{w} = 0$ (nonostante appartenga a Λ), il CVP si riduce all'SVP.

Osservazione 1.2.2. La soluzione ad entrambi i problemi in generale non è unica: ad esempio $(1,0)$, $(-1,0)$, $(0,1)$, $(0,-1)$ risolvono l'SVP in $\mathbb{Z}^2$, mentre $(0,0)$,$(1,0)$,$(1,1)$,$(0,1)$ risolvono il CVP in $\mathbb{Z}^2$ con $\underline{w} = (\frac{1}{2}, \frac{1}{2})$.

Osservazione 1.2.3. Se una base $\mathcal{B} = \{\underline{v}_1, \ldots, \underline{v}_n\}$ del reticolo Λ è ortogonale, i due problemi hanno facile risoluzione, osservando che il calcolo del quadrato della lunghezza di un vettore generico del reticolo

$$\underline{v} = \alpha_1\underline{v}_1 + \cdots + \alpha_n\underline{v}_n$$

si riduce a:

$$\|\underline{v}\|_2^2 = \|\alpha_1\underline{v}_1 + \cdots + \alpha_n\underline{v}_n\|_2^2 = \alpha_1^2\|\underline{v}_1\|_2^2 + \cdots + \alpha_n^2\|\underline{v}_n\|_2^2$$

SVP I vettori di lunghezza minima di Λ sono i generatori di norma minore e i loro opposti.

CVP Il vettore $\underline{w}$ si può scrivere come combinazione lineare a coefficienti in $\mathbb{R}$ dei vettori della base

$$\underline{w} = s_1\underline{v}_1 + \cdots + s_n\underline{v}_n$$

da cui

$$\underline{v} - \underline{w} = (\alpha_1 - s_1)\underline{v}_1 + \cdots + (\alpha_n - s_n)\underline{v}_n$$
$$\|\underline{v} - \underline{w}\|_2^2 = (\alpha_1 - s_1)^2\|\underline{v}_1\|_2^2 + \cdots + (\alpha_n - s_n)^2\|\underline{v}_n\|_2^2$$

Quindi è sufficiente scegliere $\alpha_i = \lfloor t_i \rceil$ per ogni i. Vi sono più soluzioni quando $\alpha_i - \lfloor \alpha_i \rfloor = \frac{1}{2}$ per qualche i.

1.2.1 Stima teorica della lunghezza del vettore più corto

Ci si pone l'obiettivo di stimare dal punto di vista teorico la lunghezza minima di un vettore non nullo di Λ, ossia un vettore che risolva SVP.

Teorema 1.2.4 (Minkowski)**.** *Sia Λ un reticolo di dimensione n e sia S un sottoinsieme misurabile (rispetto alla misura di Lebesgue) di $\mathbb{R}^n$ tale che la sua misura sia maggiore del determinate di Λ, cioè $\mu(S) > \det(\Lambda)$. Allora esistono due elementi distinti di S $\underline{x}$ e $\underline{y}$ la cui differenza appartiene al reticolo.*

Dimostrazione. Sia $\mathcal{B}$ una base di Λ e sia $\mathscr{F}_{\mathcal{B}}$ il dominio fondamentale associato a tale base. S può essere scritto come unione disgiunta di S intersecato alle traslazioni di $\mathscr{F}_{\mathcal{B}}$ mediante ogni elemento del reticolo, grazie al teorema 1.1.5.

$$S = \bigsqcup_{v \in \Lambda} \left[S \cap \left(\underline{v} + \mathscr{F}_{\mathcal{B}} \right) \right] \tag{1.23}$$

L'unione è disgiunta e numerabile, perciò

$$\mu(S) \overset{(1.23)}{=} \mu \left(\bigsqcup_{\underline{v} \in \Lambda} \left[S \cap \left(\underline{v} + \mathscr{F}_{\mathcal{B}} \right) \right] \right) \overset{\sigma\text{-additività}}{=} \sum_{\underline{v} \in \Lambda} \mu \left(S \cap \left(\underline{v} + \mathscr{F}_{\mathcal{B}} \right) \right). \tag{1.24}$$

Poiché la misura di Lebesgue μ è invariante per traslazioni

$$\mu \left(S \cap \left(\underline{v} + \mathscr{F}_{\mathcal{B}} \right) \right) = \mu \left((S - \underline{v}) \cap \mathscr{F}_{\mathcal{B}} \right). \tag{1.25}$$

Se tali insiemi fossero disgiunti si avrebbe che

$$\det(\Lambda) \overset{\text{def}}{=} \mu \left(\mathscr{F}_{\mathcal{B}} \right) \geq \sum_{\underline{v} \in \Lambda} \mu \left((S - \underline{v}) \cap \mathscr{F}_{\mathcal{B}} \right) \overset{(1.25)}{=} \sum_{\underline{v} \in \Lambda} \mu \left(S \cap \left(\underline{v} + \mathscr{F}_{\mathcal{B}} \right) \right) \overset{(1.24)}{=} \mu(S)$$

il che contraddice l'ipotesi. Ciò significa che esistono due elementi $\underline{v}$ e $\underline{w}$ distinti nel reticolo Λ tali che

$$\left((-\underline{v} + S) \cap \mathscr{F}_{\mathcal{B}} \right) \bigcap \left((-\underline{w} + S) \cap \mathscr{F}_{\mathcal{B}} \right) \neq \emptyset.$$

In particolare $(-\underline{v} + S) \cap (-\underline{w} + S) \neq \emptyset$, perciò esistono $\underline{x}$ e $\underline{y}$ appartenenti a S tali che

$$-\underline{v} + \underline{x} = -\underline{w} + \underline{y},$$

riscrivibile come

$$\underbrace{\underline{x} - \underline{y}}_{\in S} = \underbrace{\underline{w} - \underline{v}}_{\neq 0} \in \Lambda,$$

da cui la tesi. □

Corollario 1.2.5. *Sia Λ un reticolo di dimensione n e sia S un sottoinsieme di $\mathbb{R}^n$ misurabile, convesso e simmetrico rispetto all'origine. Se $\mu(S) > 2^n \det(\Lambda)$, allora esiste un elemento non nullo di S appartenente a Λ. Se S è anche compatto, la tesi vale anche quando $\mu(S) = 2^n \det(\Lambda)$.*

Dimostrazione. Per la prima parte dell'enunciato, è sufficiente applicare il teorema di Minkowski 1.2.4 a $\frac{1}{2}S$ e sfruttare le ipotesi su S. Per la seconda parte, si applica quanto appena dimostrato a $(1+\epsilon)S$ per ogni $\epsilon > 0$ e si interseca al variare di ϵ. Per più dettagli si rimanda a [14]. □

Teorema 1.2.6 (Hermite)**.** *Ogni reticolo Λ di dimensione n contiene un vettore $\underline{v}$ che soddisfi*

$$\|\underline{v}\| \leq \sqrt{n} \det(\Lambda)^{1/n} \tag{1.26}$$

Dimostrazione. Si applica il corollario al teorema di Minkowski 1.2.5 scegliendo come S l'ipercubo di $\mathbb{R}^n$ di lato $2\det(\Lambda)^{1/n}$, centrato nell'origine, affinché ogni suo punto $\underline{x}$ sia tale che

$$-\det(\Lambda)^{1/n} \leq x_i \leq \det(\Lambda)^{1/n} \tag{1.27}$$

per ogni i da 1 a n, indicando con x_i le coordinate di $\underline{x}$.

S è convesso, simmetrico rispetto all'origine e compatto; inoltre $\mu(S) = 2^n \det(\Lambda)$. Allora esiste un elemento non nullo $\underline{v}$ di S appartenente al reticolo Λ, e grazie a (1.27)

$$\|\underline{v}\|_2 = \left(v_1^2 + \dots v_n^2\right)^{1/2} \leq \sqrt{n} \det(\Lambda)^{1/n}$$

□

La stima data dal teorema di Hermite è migliorabile applicando il teorema di Minkowski 1.2.5 ad un'ipersfera di $\mathbb{R}^n$, anziché ad un ipercubo. Poiché si è interessati a questa stima quando n è molto grande, il volume di un'ipersfera è approssimabile in questo modo:

$$\mu(B_R(\underline{x}))^{1/n} \approx \sqrt{\frac{2\pi e}{n}} R \tag{1.28}$$

dove $B_R(\underline{x})$ indica una palla di centro $\underline{x}$ e raggio R in $\mathbb{R}^n$. Per più dettagli si rimanda a [20]. Ricalcando la dimostrazione precedente, se si sceglie R tale che $\mu(B_R(0)) = 2^n \det(\Lambda)$, allora tale palla contiene un punto non nullo del reticolo, grazie al teorema 1.2.5. Usando l'approssimazione (1.28), R deve all'incirca verificare

$$\sqrt{\frac{2\pi e}{n}} R \approx 2 \det(\Lambda)^{1/n},$$

il che implica che

$$R \approx \sqrt{\frac{2n}{\pi e}} \det(\Lambda)^{1/n} \tag{1.29}$$

Quindi esiste un punto $\underline{v}$ del reticolo tale che

$$\|\underline{v}\|_2 \leq R \overset{(1.29)}{\approx} \sqrt{\frac{2n}{\pi e}} \det(\Lambda)^{1/n} \tag{1.30}$$

Poiché la cardinalità dei punti della palla $B_R(\underline{0})$ intersecata con il reticolo Λ è approssimativamente il numero di copie di domini fondamentali traslati contenuti in $B_R(\underline{0})$, allora tale quantità è circa uguale al rapporto tra la misura di tale palla e la misura di un dominio fondamentale, ossia il determinante del reticolo:

$$\#\{\underline{v} \in \Lambda : \|\underline{v}\|_2 \leq R\} \approx \frac{\mu(B_R(\underline{0}))}{\det(\Lambda)} \tag{1.31}$$

Questa stima vale anche se n è grande, a patto che R non sia troppo grande (a causa dall'errore generato dai punti del reticolo sul bordo della palla). Si desidera determinare R affinché tale rapporto sia uguale a 1, in modo da trovare il vettore più corto del reticolo. Usando nuovamente la stima (1.28),

$$1 \approx \frac{\left(\sqrt{\frac{2\pi e}{n}} R\right)^n}{\det(\Lambda)}$$

Da cui si ottiene che

$$R \approx (\det(\Lambda))^{1/n} \sqrt{\frac{n}{2\pi e}} =: \sigma_\Lambda. \tag{1.32}$$

Tale valore si dice **Euristica Gaussiana**; si tratta della stima della lunghezza minore di un elemento del reticolo Λ. La stessa euristica si ottiene anche per il CVP.

1.2.2 Algoritmo Round-Off di Babai

L'algoritmo più semplice per risolvere CVP (e quindi anche SVP) è replicare la procedura illustrata nel caso in cui la base sia ortogonale con la base che si ha a disposizione, ed è un'idea dovuta a Babai.

Il problema è che la correttezza del risultato dipende dalla qualità della base a disposizione. La spiegazione geometrica di ciò risiede nel fatto che esiste un unico $\underline{v}$ elemento di Λ per cui il vettore $\underline{w}$ appartiene al traslato di $\underline{v}$ del dominio fondamentale $\mathscr{F}$ rispetto alla base data. Quindi un buon candidato per risolvere il CVP è il vertice più vicino a $\underline{w}$ di $\mathscr{F} + \underline{v}$. Nel caso in cui i vettori siano quasi ortogonali questo funziona, altrimenti vi sono punti del reticoli fuori dal parallelepipedo che sono più vicini a $\underline{w}$ dei vertici. Si illustra quanto detto in concreto con il seguente esempio su $\mathbb{R}^2$.

Esempio 1.2.7. Sia Λ il reticolo generato da $\mathcal{B} = \{\underline{v}_1, \underline{v}_2\}$, dove $\underline{v}_1 = (5, 1)$ e $\underline{v}_2 = (-2, 8)$.

$$\|\underline{v}_1\|_2 = \sqrt{26} \quad \|\underline{v}_2\|_2 = \sqrt{68}$$

Algorithm 2 Algoritmo Round-Off di Babai

Input: $\mathbf{B} = \{\underline{v}_1, \ldots, \underline{v}_n\}$, $\underline{w}$
Output: $\underline{v}$ soluzione di apprCVP

$\underline{s} \leftarrow \underline{w}B^{-1}$
$\underline{v} \leftarrow 0$
for i=1, ..., n **do**
  $\alpha_i \leftarrow \lfloor s_i \rceil$
  $\underline{v} \leftarrow \underline{v} + \alpha_i \underline{v}_i$
end for
return $\underline{v}$

Si risolve con l'algoritmo presentato il CVP per $\underline{w} = (27, 8)$.
La basis matrix di questa base è

$$B = \begin{bmatrix} 5 & 1 \\ -2 & 8 \end{bmatrix} \qquad \det(B) = 42$$

Si procede con l'algoritmo. Si calcola dapprima l'inversa della basis matrix, per poter esprimere $\underline{W}$ in funzione di tale base.

$$B^{-1} = \frac{1}{\det(B)} \begin{bmatrix} 8 & -1 \\ 2 & 5 \end{bmatrix} = \frac{1}{42} \begin{bmatrix} 8 & -1 \\ 2 & 5 \end{bmatrix}$$

$$\underline{s} = \underline{w}B^{-1} = (27, 8) \cdot \frac{1}{42} \begin{bmatrix} 8 & -1 \\ 2 & 5 \end{bmatrix} =$$

$$= \left(\frac{27 \cdot 8 + 8 \cdot 2}{42}, \frac{-27 + 8 \cdot 5}{42} \right) = \left(5 + \frac{11}{21}, \frac{13}{42} \right)$$

Successivamente, si arrotondano i coefficienti ottenuti per determinare il vettore più vicino.

$$\underline{v} = \left\lfloor 5 + \frac{11}{21} \right\rceil \underline{v}_1 + \left\lfloor \frac{13}{42} \right\rceil \underline{v}_2 = 6\underline{v}_1 = (30, 6)$$

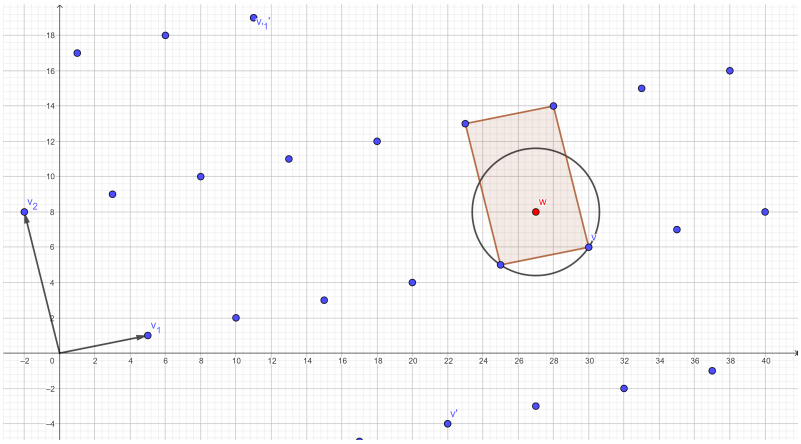
La distanza tra i due vettori è la seguente:

$$\|\underline{v} - \underline{w}\| = \|(30, 6) - (27, 8)\| = \|(3, -2)\| = \sqrt{9 + 4} = \sqrt{13} \approx 3.6$$

La figura sottostante rappresenta la situazione in esame: i vettori $\underline{v}_1$ e $\underline{v}_2$ appaiono circa ortogonali e viene confermata l'intuizione geometrica: l'output dell'algoritmo è uno dei vertici più vicini a $\underline{w}$ del dominio fondamentale traslato che lo contiene. Si sottolinea la non unicità della soluzione, e il fatto che l'algoritmo restituisce uno solo dei due vertici più vicini.

In effetti l'Hadamard Ratio di questa base è:

$$\mathcal{H}(\mathcal{B}) = \left(\frac{\det(\Lambda)}{\|\underline{v}_1\|_2 \|\underline{v}_2\|_2} \right)^{\frac{1}{2}} = \left(\frac{\det(B)}{\sqrt{26} \cdot \sqrt{68}} \right)^{\frac{1}{2}} = \left(\frac{42}{2\sqrt{442}} \right)^{\frac{1}{2}} \approx 0.999$$



La situazione cambia drasticamente utilizzando un'altra base, ottenuta a partire dalla prima mediante il prodotto della basis matrix B con una matrice P a entrate intere e determinante 1.

$$P = \begin{bmatrix} 7 & 6 \\ 8 & 7 \end{bmatrix} \qquad \det(P) = 7 \cdot 7 - 6 \cdot 8 = 1$$

$$B' = PB = \begin{bmatrix} 7 & 6 \\ 8 & 7 \end{bmatrix} \cdot \begin{bmatrix} 5 & 1 \\ -2 & 8 \end{bmatrix} = \begin{bmatrix} 23 & 55 \\ 26 & 64 \end{bmatrix}$$

$$\det(B') = \det(PB) = \det(P)\det(B) = 42$$

La nuova base ottenuta è $\mathcal{B}' = \{\underline{v}'_1, \underline{v}'_2\}$, con $\underline{v}'_1 = (23, 55)$ e $\underline{v}'_2 = (26, 64)$.

$$\|\underline{v}'_1\|_2 = \sqrt{3554} \qquad \|\underline{v}'_2\|_2 = \sqrt{4772}$$

Si performa nuovamente l'algoritmo.

$$B'^{-1} = \frac{1}{\det(B')} \begin{bmatrix} 64 & -55 \\ -26 & 23 \end{bmatrix} = \frac{1}{42} \begin{bmatrix} 64 & -55 \\ -26 & 23 \end{bmatrix}$$

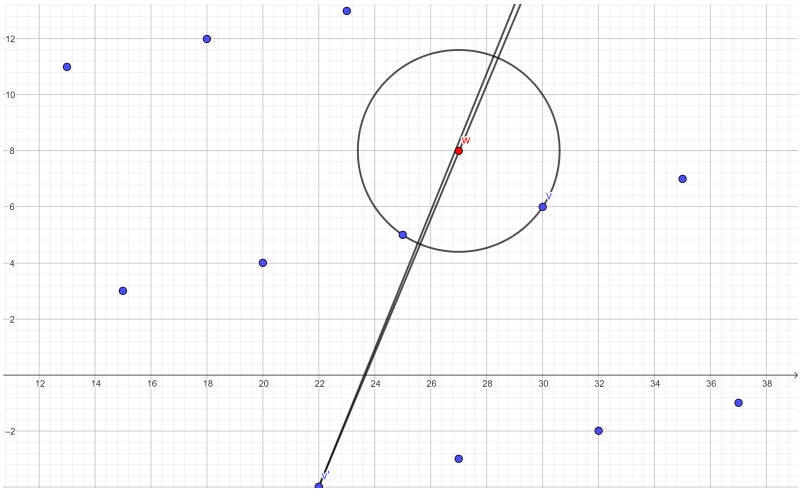
$$\underline{s} = \underline{w}B'^{-1} = (27, 8) \cdot \frac{1}{42} \begin{bmatrix} 64 & -55 \\ -26 & 23 \end{bmatrix} =$$

$$= \left(\frac{27 \cdot 64 + 8 \cdot (-26)}{42}, \frac{27 \cdot (-55) + 8 \cdot 23}{42} \right) = \left(36 + \frac{4}{21}, -30 - \frac{41}{42} \right)$$

$$\underline{v}' = \left\lfloor 36 + \frac{4}{21} \right\rceil \underline{v}_1 + \left\lfloor -30 - \frac{41}{42} \right\rceil \underline{v}_2 = 36\underline{v}_1 - 31\underline{v}_2 = (22, -4)$$

$$\|\underline{v}' - \underline{w}\|_2 = \|(22, -4) - (27, 8)\|_2 = \|(-5, -12)\|_2 = \sqrt{25 + 144} = \sqrt{169} = 13$$

Risulta chiaro che il risultato è sbagliato perché la distanza è addirittura il quadrato di quella ottenuta con l'altra base, come si vede dalla figura.



Viene rappresentato il dominio fondamentale traslato di $\underline{v}'$, che contiene $\underline{w}$; esso è un parallelogramma avente lati quasi paralleli. Il vertice più vicino a $\underline{w}$ del parallelogramma è in effetti $\underline{v}'$, ma vi sono altri punti del reticolo più vicini a $\underline{w}$. Infatti, l'Hadamard Ratio di questa base è

$$\mathcal{H}(\mathcal{B}') = \left(\frac{\det(\Lambda)}{\|\underline{v}_1'\|_2 \|\underline{v}_2'\|_2} \right)^{\frac{1}{2}} = \left(\frac{42}{\sqrt{3554} \cdot \sqrt{4772}} \right)^{\frac{1}{2}} \approx 0.1$$

Capitolo 2

Crittosistemi Lattice-Based

Durante gli anni Novanta, furono introdotti numerosi crittosistemi basati su problemi sui reticoli. I più significativi sono GGH e NTRU. Ad oggi la sicurezza di questi crittosistemi non è totalmente compresa, visto che sono più recenti degli altri crittosistemi a chiave pubblica attualmente in uso. Ciò che è certo è che questi ultimi verranno violati quando sarà costruito un computer quantistico, mentre si suppone che ciò non avvenga per i crittosistemi cosiddetti *Lattice-Based*. Inoltre, questi offrono vantaggi in termini di velocità.

Innanzitutto, si specifica formalmente cosa sia un crittosistema:

Definizione 2.0.1. *Un* ***crittosistema*** *è una ottupla* $(\mathcal{A}, \mathcal{B}, \mathcal{M}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D}, \mathcal{S})$ *in cui:*

$\mathcal{A}$ *è l'alfabeto delle parole in chiaro (plaintexts).*

$\mathcal{B}$ *è l'alfabeto delle parole cifrate (ciphertexts).*

$\mathcal{M} \subseteq \bigcup_{n\geq 1} \mathcal{A}^n$ *è l'insieme delle parole in chiaro.*

$\mathcal{C} \subseteq \bigcup_{n\geq 1} \mathcal{B}^n$ *è l'insieme delle parole cifrate.*

$\mathcal{K}$ *è l'insieme delle chiavi.*

$\mathcal{E}$ *è la funzione di cifratura (encryption),* $\mathcal{E} : \mathcal{K} \times \mathcal{M} \to \mathcal{C}$.

$\mathcal{D}$ *è la funzione di decifratura (decryption),* $\mathcal{D} : \mathcal{K} \times \mathcal{C} \to \mathcal{M}$.

$\mathcal{S} = \{(k, k') \in \mathcal{K} \times \mathcal{K} : \mathcal{D}(k', \mathcal{E}(k, x)) = x \text{ per ogni } x \in \mathcal{M}\}$

Un crittosistema si dice ***simmetrico*** *o* ***a chiave segreta*** *se* $\mathcal{S}$ *consiste nella diagonale di* $\mathcal{K} \times \mathcal{K}$*, altrimenti si dice* ***asimmetrico*** *o* ***a chiave pubblica****. In tal caso si distinguono la chiave pubblica (per cifrare messaggi) da quella privata (per decifrarli).*

Si chiamano Alice e Bob i due interlocutori di un crittosistema, Eve l'attaccante.

GGH e NTRU sono entrambi crittosistemi a chiave pubblica.

2.1 GGH

Questo sistema fu ideato da Goldreich, Goldwasser e Halevi nel 1997. GGH è un'applicazione diretta delle idee esposte nel capitolo precedente: Alice sceglie come chiave privata una buona base di un reticolo e come chiave pubblica una base cattiva dello stesso reticolo. Il messaggio di Bob è un vettore le cui entrate vengono utilizzate come coefficienti di una combinazione lineare dei vettori della base pubblica, cui viene sommato un termine di errore. Poiché Alice possiede una buona base del reticolo, può eseguire l'algoritmo di Babai per ottenere il messaggio originale. La sua difficoltà risiede nel risolvere CVP. Fu congetturato che questo problema fosse intrattabile per dimensione del reticolo maggiore di 300; in realtà vi sono algoritmi che risolvono il problema per dimensioni superiori a 350.

Si veda ora la sua specificazione nel dettaglio.

Creazione chiave privata

Alice sceglie come dimensione $n > 350$. Alice sceglie n vettori linearmente indipendenti e ragionevolmente ortogonali $\underline{v}_1, \ldots, \underline{v}_n$, che compongono la base del reticolo Λ. Una maniera suggerita di fare ciò è fissare un parametro intero positivo e scegliere casualmente le entrate di ogni vettore intere e di modulo minore di tale parametro, un'altra è scegliere come basis matrix del reticolo un multiplo della matrice identità sommato ad una matrice di perturbazione con entrate piccole.

Alice può controllare di aver scelto una buona base calcolando l'Hadamard Ratio, se così non è ne sceglie una nuova.

La chiave privata è data da $\mathcal{B}_{good} = \{\underline{v}_1, \ldots, \underline{v}_n\}$, con $\mathcal{H}(\mathcal{B}_{good}) \approx 1$.

Creazione chiave pubblica

In seguito Alice sceglie una matrice U a entrate intere e determinante ± 1 e la moltiplica alla basis matrix di $\mathcal{B}_{good}$ ottenendo una nuova basis matrix per Λ. U può essere ottenuta come prodotto di matrici elementari scelte casualmente. Le righe $\underline{v}'_1, \ldots, \underline{v}'_n$ della nuova basis matrix ottenuta costituiscono una nuova base per Λ. L'obiettivo è che questa sia una pessima base, quindi l'Hadamard Ratio deve essere molto piccolo, se così non è Alice cambia matrice U.

La chiave pubblica è data da $\mathcal{B}_{bad} = \{\underline{v}'_1, \ldots, \underline{v}'_n\}$, con $\mathcal{H}(\mathcal{B}_{bad}) \ll 1$.

Alice pubblica $\mathcal{B}_{bad}$, affinché chiunque possa inviarle messaggi da decifrare.

Cifratura

Lo spazio dei messaggi, sia in chiaro che cifrati, è $\mathbb{Z}^n$. Bob sceglie un elemento di tale insieme e sceglie un vettore di perturbazione $\underline{e}$ con entrate comprese tra $-\delta$ e δ, dove δ è un numero reale positivo, piccolo. Scrive una combinazione lineari dei vettori di $\mathcal{B}_{bad}$ con coefficienti le componenti del messaggio $\underline{m}$ e vi somma la perturbazione $\underline{e}$.

La funzione di cifratura è

$$\mathcal{E}_{\mathcal{B}_{bad}} : \mathbb{Z}^n \to \mathbb{Z}^n$$
$$\mathcal{E}_{\mathcal{B}_{bad}}(\underline{m}) = \sum_{i=1}^{n} m_i \underline{v}'_i + \underline{e} =: \underline{c}$$

dove m_i sono le componenti del messaggio in chiaro $\underline{m}$. Bob invia $\underline{c}$ ad Alice.

Osservazione 2.1.1. GGH è un crittosistema probabilistico perché un singolo messaggio in chiaro dà diversi messaggi cifrati, a seconda della perturbazione scelta. Ciò è pericoloso se si manda più volte lo stesso messaggio con perturbazioni diverse e messaggi diversi con stessa perturbazione.

Decifratura

Alice riceve $\underline{c}$ e, essendo l'unica a possedere una buona base del reticolo $\mathcal{B}_{good}$, è l'unica a poter performare l'algoritmo di Babai e recuperare il messaggio originale. Infatti $\sum_{i=1}^{n} m_i \underline{v}'_i$ è un elemento del reticolo Λ, e, a patto che $\underline{e}$ abbia entrate piccole, risolve il CVP scegliendo con target $\underline{c}$. Si ottiene $\sum_{i=1}^{n} m_i \underline{v}'_i$, non resta quindi che moltiplicare per l'inversa della matrice avente righe $\mathcal{B}_{bad}$, che si indica con B_{bad}^{-1}, per ottenere $\underline{m}$.

La funzione di decifratura è

$$\mathcal{D}_{\mathcal{B}_{good}} : \mathbb{Z}^n \to \mathbb{Z}^n$$
$$\mathcal{D}_{\mathcal{B}_{good}}(\underline{c}) = \left\lfloor \underline{c} B_{good}^{-1} \right\rceil B_{good} B_{bad}^{-1}$$

Osservazione 2.1.2. Se Eve riesce ad intercettare il messaggio cifrato $\underline{c}$, può usare l'algoritmo di Babai solo con $\mathcal{B}_{bad}$, il che porta ad un risultato fuorviante per come è stata scelta tale base.

Esempio 2.1.3. Sia $\mathcal{B}_{good} = \{\underline{v}_1, \underline{v}_2\}$ la base di un reticolo Λ di dimensione 2, con

$$\underline{v}_1 = (1, 45) \qquad \underline{v}_2 = (45, -1).$$

Si tratta di una buona base perché i due vettori sono ortogonali, il che significa che corrispondono alla propria ortogonalizzazione, allora

$$\det(\Lambda) = \left|\det \begin{bmatrix} 1 & 45 \\ 45 & -1 \end{bmatrix}\right| = |-1-45^2| = 2026$$

e perciò $\mathcal{H}(\mathcal{B}_{good}) = 1$.

Alice prende

$$P = \begin{bmatrix} 7 & 8 \\ 6 & 7 \end{bmatrix}$$

il cui determinante è 1, e la moltiplica per la basis matrix associata a $\mathcal{B}_{good}$, ottenendo la basis matrix per la nuova base $\mathcal{B}_{bad}$:

$$B_{bad} = PB_{good} = \begin{bmatrix} 7 & 8 \\ 6 & 7 \end{bmatrix} \begin{bmatrix} 1 & 45 \\ 45 & -1 \end{bmatrix} = \begin{bmatrix} 367 & 307 \\ 321 & 263 \end{bmatrix}$$

Da cui si ottiene $\mathcal{B}_{bad} = \{\underline{v}'_1, \underline{v}'_2\}$, con

$$\underline{v}'_1 = (367, 307) \qquad \underline{v}'_2 = (321, 263).$$

Si calcola l'Hadamard Ratio della nuova base:

$$\mathcal{H}(\mathcal{B}_{bad}) = \left(\frac{\det(\Lambda)}{\|\underline{v}'_1\|\|\underline{v}'_2\|}\right)^{\frac{1}{2}} = \left(\frac{2026}{\sqrt{228938}\cdot\sqrt{172210}}\right)^{\frac{1}{2}} \approx 0.101$$

Alice pubblica $\mathcal{B}_{bad}$.

Bob sceglie $\underline{m} = (35, 27)$ messaggio in chiaro e perturbazione $\underline{e} = (-9, 2)$. Bob calcola la cifratura del suo messaggio:

$$\underline{c} = 35\underline{v}'_1 + 27\underline{v}'_2 + \underline{e} = 35(367, 307) + 27(321, 263) + (-9, 2) = (21503, 17848)$$

Bob invia $\underline{c}$ ad Alice. Alice riceve $\underline{c}$ ed applica ad esso l'algoritmo di Babai utilizzando $\mathcal{B}_{good}$.

$$\underline{t} = \underline{c}B^{-1}_{good} = (21503, 17848) \cdot \frac{1}{-2026}\begin{bmatrix} -1 & -45 \\ -45 & 1 \end{bmatrix} \approx (407.04, 468.799)$$

$$\underline{a} = (\lfloor t_1 \rceil, \lfloor t_2 \rceil) = (407, 469)$$

Allora Alice può calcolare il vettore più vicino a $\underline{c}$ del reticolo

$$\underline{v} = a_1\underline{v}_1 + a_2\underline{v}_2 = 407(1, 45) + 469(45, -1) = (21512, 17846)$$

da cui deduce che

$$\underline{e} = \underline{c} - \underline{v} = (21503, 17848) - (21512, 17846) = (-9, 2).$$

Inoltre, può recuperare il messaggio $\underline{m}$ moltiplicando $\underline{v}$ per l'inversa della matrice B_{bad}.

$$\underline{m} = \underline{v}B_{bad}^{-1} = (21503, 17846) \cdot \frac{1}{-2026}\begin{bmatrix} 319 & -40 \\ -1853 & 226 \end{bmatrix} = (35, 27).$$

Se Eve prova a utilizzare l'algoritmo Babai con $\mathcal{B}_{bad}$ ottiene:

$$\underline{t}' = \underline{c}B_{bad}^{-1} = (21503, 17848) \cdot \frac{1}{-2026}\begin{bmatrix} 319 & -40 \\ -1853 & 226 \end{bmatrix} \approx (36.485, 25.274)$$

da cui

$$\underline{a}' = (\lfloor t_1' \rceil, \lfloor t_2' \rceil) = (36, 25)$$
$$\underline{v}' = \underline{a}'B_{bad} = a_1'\underline{v}_1' + a_2'\underline{v}_2' = 36(367, 307) + 25(321, 263) = (21237, 17627) \neq \underline{v}$$
$$\underline{m}' = \underline{v}'B_{bad}^{-1} = \underline{a}'B_{bad}B_{bad}^{-1} = \underline{a}' = (36, 25) \neq \underline{m}$$

Osservazione 2.1.4. Micciancio propose una variante di GGH, in cui si scambiano i ruoli di e e m, ossia si cifra il messaggio come perturbazione ed e diventa il vettore dei coefficienti di un elemento del reticolo. Attraverso questa costruzione, si può dire che NTRU è un caso speciale della variante di Micciancio di GGH.

2.2 NTRU

NTRU fu presentato per la prima volta nel 1996 alla conferenza annuale internazionale di crittografia *Crypto 96*, da Hoffstein, Pipher e Silverman. Esso fa uso di quozienti di anelli di polinomi, ma il problema di fondo può essere visto come un SVP o un CVP, a seconda che si provi a recuperare la chiave o il messaggio,in una classe speciale di reticoli detti *reticoli NTRU*, di dimensione pari e contenenti tutti i vettori $(\underline{x}, \underline{y})$ tali che

$$\underline{y} = \underline{x}H \mod q, \tag{2.1}$$

dove q è un parametro intero fissato e pubblico e H è una matrice circolante, per cui è sufficiente specificarne la prima riga (ciò risulta in una dimensione della chiave pubblica minore che in GGH). H è la chiave pubblica del crittosistema. La sicurezza del crittosistema si basa sulla difficoltà di trovare vettori corti in tali reticoli.

La chiave privata è un elemento del reticolo e compone una mezza base del reticolo con altri vettori facilmente ricavabili da essa, utile per poter risolvere istanze di CVP.

Anelli di polinomi di convoluzione

Sia N un intero. L'anello dei polinomi di convoluzione di rango N è l'anello quoziente

$$R = \frac{\mathbb{Z}[x]}{x^N - 1}.$$

Similmente, l'anello dei polinomi di convoluzione di rango N modulo q (con q un intero) è l'anello quoziente

$$R_q = \frac{\mathbb{Z}_q[x]}{x^N - 1}$$

dove con $\mathbb{Z}_q$ si indica l'anello quoziente $\mathbb{Z}/q\mathbb{Z}$.

Quozientare rispetto a $x^N - 1$ equivale a ridurre modulo N gli esponenti dei monomi che compaiono in un polinomio, perciò ogni elemento di R o R_q ammette un unico rappresentante della forma

$$a_0 + a_1 x + \cdots + a_{N-1} x^{N-1}$$

con coefficienti rispettivamente in $\mathbb{Z}$ o $\mathbb{Z}_q$.

Ogni polinomio può essere identificato col vettore dei coefficienti di tale rappresentante $(a_0, \ldots, a_{N-1}) = \underline{a}$.

Dati due polinomi $a(x)$ e $b(x)$ rappresentati nella maniera descritta sopra, e i loro vettori dei coefficienti $(a_0, \ldots, a_{N-1})$ e $(b_0, \ldots, b_{N-1})$, l'addizione tra di essi corrisponde all'operazione di somma dei vettori dei coefficienti:

$$a(x) + b(x) = s(x) \quad \longleftrightarrow \quad (a_0, \ldots, a_{N-1}) + (b_0, \ldots, b_{N-1}) = (s_0, \ldots, s_{N-1}).$$

Per quanto riguarda la moltiplicazione, ci si deve ricordare di ridurre nuovamente modulo N le potenze dei monomi ottenuti, da cui

$$a(x) \star b(x) = p(x) \quad \longleftrightarrow \quad (a_0, \ldots, a_{N-1}) \star (b_0, \ldots, b_{N-1}) = (p_0, \ldots, p_{N-1})$$

con $p_k = \sum_{i+j=k \mod N} a_i b_j$ per ogni k.

Si indica tale prodotto con $\star$ perché sui vettori delle componenti corrisponde all'operazione di convoluzione.

Definizione 2.2.1. *Dati d_1 e d_2 interi positivi, si dicono* ***polinomi ternari*** *con parametri d_1 e d_2 gli elementi dell'insieme*

$$\mathcal{T}(d_1, d_2) = \left\{ \begin{array}{r} d_1 \text{ coefficienti uguali a 1,} \\ a(x) \in R : a(x) \text{ ha } d_2 \text{ coefficienti uguali a } -1 \\ \text{e i restanti uguali a 0} \end{array} \right\}$$

Analogamente, si dicono ***polinomi binari*** *quelli aventi coefficienti unicamente uguali a 0 o 1.*

Osservazione 2.2.2. Il prodotto $\underline{a} \star \underline{b}$ richiede in generale N^2 moltiplicazioni. Quando uno dei due polinomi è ternario, questo prodotto si può calcolare senza fare moltiplicazioni, richiedendo approssimativamente $O(N^2)$ addizioni e sottrazioni, riducendo di molto il costo computazionale.

Vi è un modo naturale di passare da R a R_q, che consiste nel ridurre i coefficienti modulo q. Tale riduzione è un omomorfismo di anelli perché commuta con somma e prodotto. La direzione opposta è percorribile in più modi, tra cui si privilegia il **sollevamento centrato**: dato $a(x) \in R_q$, si tratta dell'unico polinomio $a'(x) \in R$ tale che sia un rappresentante di $a(x)$ modulo q con coefficienti compresi tra $-\frac{q}{2}+1$ e $\frac{q}{2}$. Esso non è un omomorfismo di anelli.

Proposizione 2.2.3. *Sia q un numero primo. Un elemento $a(x)$ di R_q ammette inverso moltiplicativo se e solo se* $\gcd(a(x), x^N-1) = 1$ *in $\mathbb{Z}_q[x]$. Tale inverso può essere calcolato mediante l'Algoritmo Euclideo Esteso, infatti soddisfa la relazione*

$$a(x)a(x)^{-1} + (x^N - 1)v(x) = 1.$$

NTRUEncrypt

NTRUEncrypt è il crittosistema a chiave pubblica NTRU. Si tratta di un crittosistema ring-based, perché coinvolge entrambe le operazioni di anello, a differenza di crittosistemi come RSA, Diffie-Hellmann, che si dicono group-based, perché coinvolgono un'unica operazione di gruppo. Si presenta una sua specificazione rigorosa.

Parametri pubblici

Alice sceglie innanzitutto N un primo e due moduli p e q, il primo più piccolo del secondo. Le relazioni che devono essere soddisfatte sono

$$\gcd(N, q) = 1 \tag{2.2}$$
$$\gcd(p, q) = 1 \tag{2.3}$$

Scelte frequenti di questi parametri sono $N = 401$, $p = 3$ e $q = 2048$. Inoltre, viene scelto un intero piccolo d che specifica il legame tra p e q:

$$q > p(6d + 1) \tag{2.4}$$

Questi parametri vengono resi pubblici da Alice. Gli anelli in cui ha luogo il crittosistema sono:

$$R = \frac{\mathbb{Z}[x]}{x^N - 1} \qquad R_q = \frac{\mathbb{Z}_q[x]}{x^N - 1} \qquad R_p = \frac{\mathbb{Z}_p[x]}{x^N - 1}.$$

Chiave privata

Alice sceglie un polinomio $f(x)$ appartenente a $\mathcal{T}(d+1, d)$, che sia invertibile modulo p e modulo q, e un polinomio $g(x)$ appartenente a $\mathcal{T}(d, d)$. Per verificare che f sia invertibile modulo p e modulo q, è sufficiente calcolare il massimo comun divisore di $f(x)$ e $x^N - 1$ modulo p e modulo q; se esso è maggiore di 1 in almeno uno dei due casi, Alice deve scegliere un altro f. La chiave privata si compone unicamente del polinomio $f(x)$, ma $g(x)$ deve essere tenuto segreto e scartato non appena non sia più necessario, ossia dopo la creazione della chiave pubblica.

Osservazione 2.2.4. $f(x)$ viene scelto in $\mathcal{T}(d+1, d)$ perché gli elementi di $\mathcal{T}(d, d)$ non sono invertibili in R_q per ogni q. Infatti, se $a(x)$, appartenente a $\mathcal{T}(d, d)$, fosse invertibile modulo q, esisterebbe $b(x)$ in R_q tale che

$$a(x)b(x) = 1 \mod q.$$

Allora

$$a(1)b(1) = 1 \mod q$$

e quindi $a(1) \neq 0 \mod q$. Quanto provato è equivalente a dire che se $a(1) = 0 \mod q$, allora $a(x)$ non è invertibile in $\mathbb{Z}_q[x]$.

Se $a(x)$ è un elemento di $\mathcal{T}(d, d)$, ha lo stesso numero di coefficienti rispettivamente uguali a 1 e -1. Valutare il polinomio in 1 equivale a sommare tutti i coefficienti, e tale somma dà 0 per i polinomi di tale classe; quindi $a(x)$ non può essere invertibile modulo q.

Chiave pubblica

Alice può computare mediante l'Algoritmo Euclideo Esteso l'inverso di $f(x)$ modulo q, che si denota con $f_q^{-1}(x)$ e calcolare

$$h(x) = f_q(x)^{-1} \star g(x) \tag{2.5}$$

Ora Alice può disfarsi di $g(x)$ e rendere nota la chiave pubblica $h(x) \in R_q$. Infatti, chi possiede entrambe le chiavi può calcolare $g(x)$ come prodotto di esse.

Cifratura

Lo spazio dei messaggi in chiaro è l'insieme di tutti gli elementi $m(x)$ di R i cui coefficienti m_i soddisfino la seguente relazione:

$$-\frac{1}{2}p < m_i \leq \frac{1}{2}p;$$

si tratta dei sollevamenti centrati dei polinomi di R_p.

Bob sceglie inoltre un polinomio casuale $r(x)$ in $\mathcal{T}(d,d) \subseteq R$ e calcola il messaggio cifrato moltiplicando la chiave pubblica per p e per $r(x)$, sommando il messaggio $m(x)$ e riducendo infine modulo q. Si osserva che questo passaggio rende il crittosistema probabilistico.

La funzione di cifratura è

$$\begin{gathered}\mathcal{E}_h : \mathcal{M} \to \mathcal{C} \\ c(x) = \mathcal{E}_h(m(x)) = ph(x) \star r(x) + m(x) \mod q \end{gathered} \tag{2.6}$$

dove il messaggio cifrato è un elemento di R_q.

Bob invia $c(x)$ ad Alice.

Decifratura

Alice riceve $c(x)$ e dapprima calcola

$$a(x) = f(x) \star c(x) \mod q \tag{2.7}$$

Solleva $a(x)$ da R_q a R, lo riduce modulo p e determina l'inverso di $f(x)$ modulo p (indicato con f_p^{-1}) per calcolare

$$b(x) = f_p(x)^{-1} \star a(x) \mod p \tag{2.8}$$

La funzione di decifratura è

$$\begin{gathered}\mathcal{D}_{f(x)} : \mathcal{M} \to \mathcal{C} \\ \mathcal{D}_{f(x)}(c(x)) = \mathcal{S}_p\{[\mathcal{S}_q(f(x) \star c(x) \mod q) \mod p] \star f_p^{-1} \mod p\}\end{gathered}$$

indicando con $\mathcal{S}_q$ il sollevamento centrato da R_q a R, e idem $\mathcal{S}_p$.

Proposizione 2.2.5. *Se i parametri NTRU soddisfano le condizioni richieste precedentemente*

$$b(x) \equiv m(x) \mod p \tag{2.9}$$

ossia Alice recupera il messaggio originale.

Dimostrazione.

$$\begin{aligned} a(x) &\overset{(2.7)}{\equiv} f(x) \star c(x) \mod q \equiv \\ &\overset{(2.6)}{\equiv} f(x) \star (ph(x) \star r(x) + m(x)) \mod q \equiv \\ &\overset{(2.5)}{\equiv} f(x) \star [p(f_q(x)^{-1} \star g(x)) \star r(x) + m(x)] \mod q \equiv \\ &\equiv pg(x) \star r(x) + f(x) \star m(x) \mod q \end{aligned}$$

Si desidera provare che $a(x)$, visto in R, è uguale al suo sollevamento centrato; per far ciò si stima il valore massimo e minimo dei suoi coefficienti.

- $g(x)$ e $r(x)$ sono elementi di $\mathcal{T}(d, d)$, se i coefficienti uguali a 1 e quelli uguali a -1 combaciano tra loro, il valore massimo raggiungibile per un coefficiente del prodotto è $2d$.
- $f(x)$ sta in $\mathcal{T}(d+1, d)$, mentre $m(x)$ ha entrate comprese tra $-\frac{1}{2}p$ escluso e $\frac{1}{2}p$ incluso, allora il coefficiente più grande possibile di tale prodotto è $\frac{1}{2}p(2d+1)$, considerando il caso limite in cui alle entrate uguali a -1 di f viene moltiplicato $-\frac{1}{2}p$ e a quelle uguali a 1 viene moltiplicato $\frac{1}{2}p$.

Il massimo coefficiente possibile per $a(x)$ è

$$2dp + (2d+1)\frac{1}{2}p = \left(3d + \frac{1}{2}\right)p.$$

Imponendo che tale quantità sia inferiore a $\frac{1}{2}q$ si ottiene la disuguaglianza (2.4). Allo stesso modo si stima il minimo valore di ogni coefficiente di $a(x)$, ottenendo il medesimo risultato. Perciò si può considerare $a(x)$ modulo p e computare

$$\begin{aligned} b(x) &\overset{(2.8)}{\equiv} f_p^{-1} \star a(x) \mod p \equiv \\ &\overset{(2.7)}{\equiv} f_p^{-1} \star (pg(x) \star r(x) + f(x) \star m(x)) \equiv \\ &\equiv m(x) \mod p \end{aligned}$$

□

Osservazione 2.2.6. La condizione $q > (6d+1)p$ assicura che l'operazione di decifratura non fallisca. Si potrebbe scegliere un valore di q minore per avere maggiore efficienza e chiave pubblica di dimensione minore, visto che la probabilità che un coefficiente sia uguale al massimo stimato è molto bassa. Si preferisce però scegliere q grande perché fallimenti di decifratura possono dare informazioni aggiuntive all'attaccante.

Affinché $a(x) \in R_q$ sia uguale al suo sollevamento centrato in R si sottolinea che i calcoli modulo q vengono fatti considerando per ogni coefficiente un rappresentante nell'intervallo degli interi compresi tra $-\frac{q}{2}$ escluso e $\frac{q}{2}$ incluso.

Esempio 2.2.7. Si consideri $N = 7$, $p = 3$, $q = 41$ e $d = 2$. Le condizioni sui parametri sono tutte verificate: $N = 7$ è primo e

$$\begin{aligned} \gcd(N, q) &= \gcd(7, 41) = 1 \\ \gcd(p, q) &= \gcd(3, 41) = 1 \\ q = 41 > 39 &= 13 \cdot 3 = (6d + 1)p \end{aligned}$$

Alice sceglie

$$f(x) = x^6 - x^4 + x^3 + x^2 - 1,$$

polinomio ternario di $\mathcal{T}(3, 2)$, e

$$g(x) = x^6 + x^4 - x^2 - x,$$

polinomio ternario in $\mathcal{T}(2, 2)$. Poiché $\gcd(f(x), x^7 - 1) = 1$ sia modulo p che modulo q, $f(x)$ è una buona scelta e Alice può calcolare i due inversi $f_p^{-1}(x)$ e $f_q^{-1}(x)$:

$$\begin{aligned} f_q^{-1}(x) &= f(x)^{-1} \mod q = 8x^6 - 15x^5 - 10x^4 - 20x^3 - x^2 + 2x - 4 \mod 41 \\ f_p^{-1}(x) &= f(x)^{-1} \mod p = x^6 - x^5 + x^3 + x^2 + x + 1 \mod 3 \end{aligned}$$

Alice può ora determinare

$$h(x) = f_q^{-1}(x) \star g(x) \mod q = 20x^6 - x^5 + 2x^4 - 3x^3 + 8x^2 - 15x - 11 \mod 41$$

La chiave privata di Alice è $f(x) \in R$, mentre la chiave pubblica è $h(x) \in R_q$. Bob sceglie il messaggio

$$m(x) = -x^5 + x^3 + x^2 - x + 1$$

da inviare ad Alice, e sceglie come polinomio ternario casuale in $\mathcal{T}(2, 2)$

$$r(x) = x^6 - x^5 + x - 1$$

Bob calcola il messaggio cifrato

$$c(x) = pr(x) \star h(x) + m(x) = -10x^6 + 19x^5 + 4x^4 + 2x^3 - x^2 + 3x - 16 \mod 41$$

La decifratura del messaggio avviene in due step: per prima cosa, Alice calcola

$$a(x) = f(x) \star c(x) = x^6 + 10x^5 - 8x^4 - x^3 - x^2 + x + 1 \mod 41.$$

Solleva centratamente $a(x)$ a R (come si vede nella prova della proposizione 2.2.5, i coefficienti sono i medesimi, solo visti su $\mathbb{Z}$), lo riduce modulo p, ottenendo

$$a(x) = x^6 + x^5 + x^4 - x^3 - x^2 + x - 1$$

e poi computa

$$b(x) = f_p^{-1}(x) \star a(x) = 2x^5 + x^3 + x^2 + 2x + 1 \mod 3.$$

Sollevando centratamente $b(x)$ a R, Alice ottiene il messaggio in chiaro di Bob.

2.2.1 Collegamento coi reticoli

Il problema di recuperare la chiave privata di NTRUEncrypt consiste nel determinare, a partire dalla chiave pubblica, un polinomio ternario $f(x)$ tale che il suo prodotto con $h(x)$ sia ancora ternario modulo q, nonostante il secondo abbia coefficienti uniformemente distribuiti modulo q.

La soluzione a questo problema non è unica: se $(f(x), g(x))$ è soluzione, lo sono anche tutte le sue cosiddette rotazioni, ossia

$$(x^k \star f(x), x^k \star g(x)),$$

al variare di k tra 0 e $N-1$. Vengono chiamate in tal modo perché il prodotto di convoluzione con una potenza di x risulta in una rotazione ciclica dei coefficienti di numero di posizioni uguale all'esponente di x.

Con un attacco di forza bruta, l'attaccante *Eve* deve provare tutte le possibili chiavi, ossia tutti gli elementi di $\mathcal{T}(d+1, d)$. Tale insieme ha cardinalità

$$\#\mathcal{T}(d+1,d) = \binom{N}{d+1}\binom{N-d-1}{d}.$$

Essa viene massimizzata scegliendo $d \approx \frac{N}{3}$. Eve ha N possibilità di successo, perciò farà probabilmente almeno $\frac{\#\mathcal{T}(d+1,d)}{N}$ tentativi prima di trovare una soluzione.

Definizione 2.2.8. *Dato $\underline{h} = (h_0, \ldots, h_{N-1})$ un vettore in $\mathbb{Z}^N$ e q un intero positivo, si definisce il* ***reticolo NTRU*** *Λ_h associato a $\underline{h}$ il reticolo di dimensione 2N con basis matrix la matrice a blocchi*

$$B_h = \begin{bmatrix} I_N & H \\ 0_N & qI_N \end{bmatrix}$$

dove I_N indica la matrice identica $N \times N$, 0_N la matrice nulla $N \times N$ e H è una matrice circolante avente come prima riga $\underline{h}$:

$$H = \begin{bmatrix} h_0 & h_1 & \cdots & h_{N-1} \\ h_{N-1} & h_0 & \cdots & h_{N-2} \\ \cdots & \cdots & \cdots & \cdots \\ h_1 & h_2 & \cdots & h_0 \end{bmatrix}$$

Si osserva che moltiplicare un vettore a sinistra per H equivale a calcolare il prodotto di convoluzione tra tale vettore e h. Allora tale reticolo si scrive come

$$\begin{aligned} \Lambda_h &= \left\{(\underline{a}, \underline{b}) \cdot B_h : (\underline{a}, \underline{b}) \in \mathbb{Z}^{2N}\right\} = \\ &= \left\{(\underline{a}, \underline{a} \star \underline{h} - q\underline{b}) : (\underline{a}, \underline{b}) \in \mathbb{Z}^{2N}\right\} = \\ &= \left\{(\underline{x}, \underline{y}) \in \mathbb{Z}^{2N} : \underline{y} = \underline{x} \star \underline{h} \mod q\right\} = \\ &= \left\{(\underline{x}, \underline{y}) \in \mathbb{Z}^{2N} : \underline{y} = \underline{x}H \mod q\right\} \end{aligned} \tag{2.10}$$

Inoltre

$$\det(\Lambda_h) = |\det(B_h)| = q^N. \tag{2.11}$$

$\underline{h}$ può essere il vettore dei coefficienti della chiave pubblica $h(x)$ di un crittosistema NTRUEncrypt, determinato a partire dai polinomi $f(x)$ e $g(x)$. In tal caso esiste un polinomio $u(x)$ in R tale che

$$f(x) \star h(x) = g(x) + qu(x). \tag{2.12}$$

Ogni coppia di polinomi $a(x)$ e $b(x)$ appartenenti a R si identifica con il vettore dei loro coefficienti $(a_0, \ldots, a_{N-1}, b_0, \ldots, b_{N-1}) = (\underline{a}, \underline{b})$.

Allora $(\underline{f}, \underline{g})$ è un elemento del reticolo $\Lambda_{\underline{h}}$, in quanto

$$(\underline{f}, -\underline{u})B_h = (\underline{f}, \underline{g}) \tag{2.13}$$

Infatti,

$$(\underline{f}, -\underline{u})B_h = (\underline{f}, \underline{f} \star \underline{h} - q\underline{u}) \overset{(2.12)}{=} (\underline{f}, \underline{g})$$

Allo stesso modo, le rotazioni di $(\underline{f}, \underline{g})$ sono elementi del reticolo: per ogni $k = 0, \ldots, N-1$

$$(x^k \star \underline{f}, x^k \star (-\underline{u}))B_h = (x^k \star \underline{f}, (x^k \star \underline{f}) \star \underline{h} - q(x^k \star \underline{u})) = (x^k \star \underline{f}, x^k \star \underline{g}).$$

Inoltre, tali elementi hanno stessa norma di $(\underline{f}, \underline{g})$ perché le entrate sono solo permutate ciclicamente.

Dato che $f(x)$ appartiene a $\mathcal{T}(d+1, d)$ e $g(x)$ a $\mathcal{T}(d, d)$

$$\|(\underline{f}, \underline{g})\| \approx \sqrt{4d} \tag{2.14}$$

Scegliendo

$$d \approx \frac{N}{3} \tag{2.15}$$

per massimizzare il numero di possibili chiavi, si ottiene che

$$\|(\underline{f}, \underline{g})\| = O(\sqrt{N}) \text{ per } N \to \infty \tag{2.16}$$

L'Euristica Gaussiana (1.32) stima che il vettore più corto del reticolo abbia lunghezza

$$\sigma_{\Lambda_h} = \det(\Lambda_h)^{1/2N} \sqrt{\frac{2N}{2\pi e}} \overset{(2.12)}{=} \sqrt{\frac{Nq}{\pi e}} \overset{(2.4)}{>} \sqrt{\frac{N(6d+1)p}{\pi e}} \overset{(2.15)}{\approx} \sqrt{N^2 \frac{p}{\pi e}} = O(N) \tag{2.17}$$

Perciò è altamente probabile $(\underline{f}, \underline{g})$ e le sue rotazioni siano i vettori più corti del reticolo. Allora il problema di determinare la chiave privata di NTRU si basa sulla difficoltà di risolvere SVP su Λ_h.

Capitolo 3

Schemi di firma Lattice-Based

Ogni giorno si usano firme per sottoscrivere contratti, ritirare denaro in banca, concludere lettere... Ciò che una firma garantisce è la provenienza o l'autore di quanto firmato. La firma digitale replica questo processo virtualmente: un documento telematico D viene allegato di una parte addizionale con l'obiettivo di provare che tale documento appartiene o è stato creato da un certo utente. Le differenze principali tra firme analogiche e firme digitali sono che le prime sono tutte uguali e se ne verifica l'autenticità confrontandole con una firma verificata, mentre le seconde variano a seconda del documento, anche quando l'autore è lo stesso, e si verificano mediante una funzione pubblica.

Definizione 3.0.1. *Uno* ***schema di firma digitale*** *è una sestupla* $(\mathcal{D}, \mathcal{S}, \mathcal{K}, \boldsymbol{sig}, \boldsymbol{ver}, \mathcal{W})$*, in cui:*

- $\mathcal{D}$ *è l'insieme dei documenti.*
- $\mathcal{S}$ *è l'insieme delle firme.*
- $\mathcal{K}$ *è l'insieme delle chiavi.*
- $\boldsymbol{sig} : \mathcal{K} \times \mathcal{D} \to \mathcal{S}$ *è la funzione di firma.*
- $\boldsymbol{ver} : \mathcal{K} \times \mathcal{D} \times \mathcal{S} \to \{true, false\}$ *è la funzione di verifica.*
- $\mathcal{W} \subseteq \mathcal{K} \times \mathcal{K}$ *tale che per ogni coppia di chiavi* (k, k') *(la prima privata e la seconda pubblica) in* $\mathcal{W}$

$$\boldsymbol{ver}_{k'}(x, y) = \begin{cases} true & \text{se } y = \boldsymbol{sig}_k(x) \\ false & \text{altrimenti} \end{cases}$$

Una funzione di hash è $H : X \to Y$ è una funzione tale che il dominio abbia cardinalità di molto maggiore del codominio, e tale che verifichi:

1. **Preimage resistance**

   A partire da un output y di H, è difficile determinare $x \in X$ che verifichi $H(x) = y$.

2. **Second preimage resistance**

   A partire da $H(x)$, è difficile determinare un elemento $x' \in X$ diverso da x tale che $H(x) = H(x')$.

3. **Collision resistance** È difficile determinare due elementi distinti di X x' e x tali che $H(x) = H(x')$.

La maggior parte degli schemi di firma è del tipo Hash and Sign: al documento viene applicata una funzione di hash ottenendo $m = H(d)$ il cosiddetto *message digest*, cioè l'assimilazione del documento, è questo ad essere firmato, e non il documento. Si sostituisce dunque allo spazio dei documenti direttamente quello dei message digest. Deve essere computazionalmente difficile per chiunque non possieda la chiave privata firmare un documento x, si ha una **falsificazione** quando viene prodotta una firma accettabile da chiunque non sia il vero autore. La firma deve dipendere sia dall'autore che dal documento da firmare, altrimenti si potrebbe apporre una firma già usata su un nuovo documento, spacciandosi per l'autore.

In questo capitolo si trattano gli schemi di firma digitale *GGH Signature Scheme* e *NTRUSign*, ideati a partire dai crittosistemi GGH e NTRU.

3.1 Schema di firma GGH

Questo schema di firma digitale è dovuto ai medesimi autori del crittosistema GGH ed è ad esso coevo. La conversione del crittosistema GGH in uno schema di firma digitale è diretto: la chiave privata rimane una buona base $\mathcal{B}$ di un reticolo Λ, mentre la chiave pubblica è ancora una cattiva base $\mathcal{B}'$ dello stesso reticolo. Dato un message digest $\underline{d} \in \mathbb{R}^n$, la firma è il vettore dei coefficienti rispetto alla base cattiva della soluzione dell'apprCVP data dall'algoritmo Round-Off di Babai, applicato con la base segreta. La verifica consiste nel controllare se il message digest e il vettore del reticolo con coefficienti la firma sono abbastanza vicini.

Spazio dei message digest

L'insieme dei message digest $\mathcal{D}$ è $\mathbb{R}^n$ a cui vengono sottratti i punti del reticolo Λ; infatti, se il documento da firmare fosse un elemento del reticolo, chiunque potrebbe verificarne l'appartenenza al reticolo mediante la chiave pubblica (moltiplicando il vettore per l'inversa della basis matrix si otterrebbe un vettore in $\mathbb{Z}^n$) e dunque apporre la firma corretta, ossia il vettore ottenuto.

Creazione delle chiavi

Alice sceglie una buona base $\mathcal{B} = \{\underline{v}_1, \ldots, \underline{v}_n\}$ per un reticolo Λ e la tiene segreta. Così come nel crittosistema GGH, la modifica, fino ad ottenere una nuova base $\mathcal{B}' = \{\underline{w}_1, \ldots, \underline{w}_n\}$, che abbia Hadamard Ratio basso.

Firma

Alice sceglie un documento e vi applica la funzione di hash per ottenere il suo message digest $\underline{d} \in \mathcal{D}$. Poiché possiede una buona base del reticolo, può applicare con successo l'algoritmo di Babai per risolvere l'apprCVP; la firma $\underline{s}$ si ottiene moltiplicando per l'inversa della basis matrix della base pubblica. Perciò lo spazio delle firme $\mathcal{S}$ è $\mathbb{Z}^n$. La funzione di firma è:

$$\begin{aligned} \mathbf{sig}_{\mathcal{B}} &: \mathcal{D} \to \mathcal{S} \\ \mathbf{sig}_{\mathcal{B}}(\underline{d}) &= \left\lfloor \underline{d}B^{-1} \right\rceil BB'^{-1} =: \underline{s} \end{aligned} \tag{3.1}$$

dove B indica la basis matrix di $\mathcal{B}$ e B' quella associata a $\mathcal{B}'$. Quindi la firma consiste nel vettore dei coefficienti della soluzione dell'apprCVP per $\underline{d}$ rispetto alla base pubblica. Alice invia a Bob la coppia formata dal documento il cui hash è $\underline{d}$ e $\underline{s}$.

Verifica

Quando Bob riceve la coppia di documento e firma, non gli resta che calcolare $\underline{d}$ (la funzione di hash è sempre pubblica), determinare il vettore del reticolo indicato da Alice come vicino a $\underline{d}$ e verificare che la distanza tra i due sia piccola. La funzione si verifica è:

$$\begin{aligned} \mathbf{ver}_{\mathcal{B}'} &: \mathcal{D} \times \mathcal{S} \to \{true, false\} \\ \mathbf{ver}_{\mathcal{B}'}(\underline{d}, \underline{s}) &= \begin{cases} true & \text{se } \|\underline{d} - \underline{s}B'\| \text{ è piccola} \\ false & \text{altrimenti} \end{cases} \end{aligned} \tag{3.2}$$

La difficoltà di firmare un documento equivale a saper risolvere l'apprCVP rispetto a tale documento. Quando non si è in possesso di una buona base, tale operazione risulta complicata.

Esempio 3.1.1. Si può trasfromare l'esempio 2.1.3 in un esempio dello schema di firma GGH. Sia $\mathcal{B} = \{\underline{v}_1, \underline{v}_2\}$ la chiave privata e $\mathcal{B}' = \{\underline{w}_1, \underline{w}_2\}$ la chiave pubblica, dove

$$\begin{aligned} \underline{v}_1 &= (1, 45) & \underline{v}_2 &= (45, -1) \\ \underline{w}_1 &= (367, 307) & \underline{w}_2 &= (321, 263) \end{aligned}$$

Sia $\underline{d} = (21503, 17848)$ il message digest da firmare, esso non è un elemento del reticolo. Corrisponde al messaggio cifrato nell'esempio del crittosistema.

Alice computa il vettore più vicino a $\underline{d}$ del reticolo mediante l'algoritmo di Babai, ottenendo

$$\underline{v} = (21512, 17846),$$

i cui coefficienti rispetto alla base pubblica $\mathcal{B}'$ sono

$$\underline{s} = (407, 469).$$

La coppia message digest e firma è data da $(\underline{d}, \underline{s})$.

Bob calcola il vettore $\underline{v}$ moltiplicando $\underline{s}$ per la basis matrix di $\mathcal{B}'$. La distanza tra $\underline{d}$ e $\underline{v}$ è la norma della differenza cioè $\|(-9, 2)\| = \sqrt{85} \approx 9.22$, che è un valore abbastanza basso da far concludere a Bob che la firma sia corretta.

Se Eve prova a risolvere l'apprCVP con la base pubblica ottiene

$$\underline{s}' = (36, 25),$$

da cui il vettore più vicino dovrebbe essere

$$\underline{v}' = (21237, 17627)$$

La distanza tra $\underline{v}'$ e il documento è la norma del vettore $(266, 221)$, ossia $\sqrt{119597} \approx 345.828$, il che fa capire a Bob che questa non è la firma di Alice, perché tale quantità è troppo grande.

Un problema che accomuna GGH e il suo schema di firma è che, affinché l'apprCVP sia di difficile risoluzione senza la chiave privata, si considerano reticoli di dimensioni elevate, che risultano in chiavi di grandi dimensioni. Una possibile soluzione per diminuire la lunghezza della chiave pubblica è scegliere reticoli NTRU: è sufficiente la conoscenza del vettore $\underline{h}$ e del parametro q per generare la basis matrix del reticolo, ma è data solo una mezza base buona del reticolo, formata da $(\underline{f}, \underline{g})$ e le sue rotazioni; questa idea è alla base dello schema NTRUSign.

Analisi delle trascrizioni

Un altro problema è che in ogni schema di firma digitale, ogni coppia documento-firma rivela informazioni circa la chiave privata: tale documento firmato con la chiave privata fornisce tale firma. Una trascrizione sufficientemente lunga di documenti firmati può consentire di recuperare la chiave privata o saper firmare nuovi documenti. Nel caso GGH, data una coppia $(\underline{d}, \underline{s})$ di message digest e firma, per costruzione si ha che

$$\underline{d} - \underline{s}B' = \sum_{i=1}^{n} \epsilon_i(\underline{d}, \underline{s})\underline{v}_i \qquad \text{con } |\epsilon_i(\underline{d}, \underline{s})| \leq \frac{1}{2} \text{ per ogni } i = 1, \ldots, n \tag{3.3}$$

Al variare di $(\underline{d}, \underline{s})$, $\epsilon_i(\underline{d}, \underline{s})$ saranno uniformemente distribuiti in $\left[-\frac{1}{2}, \frac{1}{2}\right]$ e dunque le varie differenze saranno punti uniformemente distribuiti nel dominio fondamentale

$$\mathscr{F} = \left\{ \epsilon_1 \underline{v}_1 + \cdots + \epsilon_n \underline{v}_n : -\frac{1}{2} < \epsilon_i \leq \frac{1}{2} \text{ per ogni } i \right\} \tag{3.4}$$

generato dai vettori della chiave privata $\mathcal{B}$. Nguyen e Regev hanno mostrato in [24] che, se la dimensione del reticolo è n, con la trascrizione di n^2 firme si può recuperare $\mathcal{B}$ a partire dalla geometria di $\mathscr{F}$.

Questo attacco può diminuire di efficacia perturbando i documenti prima di firmarli, il che rende il crittosistema meno efficiente e comunque non per forza sicuro.

Il problema che sorge è il cosiddetto **Hidden Parallelepiped Problem**, che consiste nel trovare un parallelepipedo n-dimensionale o una sua approssimazione a partire da numerosi punti distribuiti uniformemente su di esso.

Esso può essere sempre ridotto ad un **Hidden Hypercube Problem**, cioè lo stesso problema, in cui il parallelepipedo viene sostituito da un cubo di lato unitario. Questo si fa calcolando la matrice di Gram della trasposta della basis matrix B, ossia $G = B^T B$; tale matrice è simmetrica e definita positiva, perciò la è anche la sua inversa G^{-1}, ciò implica che sia scomponibile col metodo di Cholesky:

$$G^{-1} = LL^T \qquad \text{con } L \text{ triangolare inferiore}$$

Moltiplicando a destra la matrice L per un campione uniforme su di un parallelepipedo fondamentale (cioè le differenze tra documenti da firmare e firme) si ottiene un campione uniforme sull'ipercubo.

Data una matrice V con entrate intere e determinante ± 1, un intero positivo k e un punto $\underline{w} \in \mathbb{R}^n$, si definisce il **momento k-esimo** di $\mathscr{F}_V$ su $\underline{w}$

$$\mathrm{mom}_{V,k}(\underline{w}) = \mathbb{E}[\langle \underline{u}, \underline{w} \rangle^k]$$

dove $\mathbb{E}$ indica il valore atteso e $\underline{u}$ un campione estratto dalla distribuzione uniforme su $\mathscr{F}_V$. Nel caso in cui si consideri un ipercubo, la matrice V è ortogonale.

Il momento quarto viene minimizzato quando calcolato nei vettori $\pm\underline{v}_1, \ldots, \pm\underline{v}_n$, ossia in modulo uguali alla righe della matrice V, e non ci sono minimi locali; allora questo problema si risolve con la minimizzazione del momento quarto. Per la minimizzazione si può usare l'algoritmo del gradiente di discesa, un metodo iterativo per raggiungere il minimo, che, partendo da un elemento random, ad ogni passo sottrae all'elemento ottenuto al passo precedente un multiplo del gradiente calcolato in tale punto. Una volta ottenuti i vettori $\underline{v}_1, \ldots, \underline{v}_n$, si moltiplica per l'inversa della matrice L, per ottenere i vettori della base segreta che generano $\mathscr{F}$. Per approfondire questo argomento si rimanda a [24].

3.2 NTRUSign

NTRUSign è uno schema di firma digitale basato sulla risoluzione di apprCVP in reticoli NTRU. Fu ideato dagli stessi creatori di NTRUEncrypt, unitamente a Whyte e Howgrave-Graham tra il 2001 e il 2003. La sua versione senza perturbazioni è rotta da analisi di trascrizione, come la firma GGH. Introducendo almeno una perturbazione, il numero di trascrizioni necessarie per rompere lo schema diviene impraticabile.

Definizione 3.2.1. *Si definisce la* ***norma centrata*** *di un vettore dei coefficienti* $\underline{a} = (a_0, \ldots, a_{N-1})$ *di un elemento di* $R = \mathbb{Z}[x]/(x^N - 1)$ *come*

$$\|\underline{a}\|_c^2 = \sum_{i=0}^{N-1}(a_i - \mu_a)^2 = \sum_{i=0}^{N-1} a_i^2 - \frac{1}{N}\left(\sum_{i=0}^{N-1} a_i\right)^2 \tag{3.5}$$

dove $\mu_a = \frac{1}{N}\sum_{i=0}^{N-1} a_i$ *è la media dei coefficienti.*

La norma centrata di un elemento in R^n *è definita come*

$$\begin{aligned}\|(\underline{a}^1, \underline{a}^2, \ldots, \underline{a}^n)\|_c^2 &= \|\underline{a}^1\|_c^2 + \|\underline{a}^2\|_c^2 + \cdots + \|\underline{a}^n\|_c^2 = \\ &= \|(a_1^1, \ldots, a_{N-1}^1)\|_c^2 + \cdots + \|(a_1^n, \ldots, a_{N-1}^n)\|_c^2\end{aligned} \tag{3.6}$$

visto che non è detto che tali elementi siano centrati nello stesso punto.

Questa norma consiste nella norma euclidea della proiezione del vettore nello spazio ortogonale a quello generato da $(1, \ldots, 1)$ in $\mathbb{R}^n$. Essa è quasi-moltiplicativa:

$$\|\underline{a}\|_c\|\underline{b}\|_c \approx \|\underline{a}\|_c\|\underline{b}\|_c \tag{3.7}$$

Parametri pubblici

Si considerino gli interi N e q, il primo determina lo spazio

$$R = \frac{\mathbb{Z}[x]}{x^N - 1}$$

e il secondo è il modulo che determina

$$R_q = \frac{\mathbb{Z}_q[x]}{x^N - 1}.$$

Si conderi inoltre un intero positivo d, che entra in gioco per la struttura delle chiavi e $\mathcal{N}$ un bound sulla norma.

Generazione delle chiavi

Siano $f(x)$ e $g(x)$ due polinomi in $\mathcal{T}(d+1,d) \subseteq R$, invertibili modulo q e tali che in $\mathbb{Q}[x]$

$$\gcd(f(x), x^N - 1) = 1 \qquad \gcd(g(x), x^N - 1) = 1$$

La chiave pubblica è data dal polinomio di R_q

$$h(x) = f(x)^{-1} \star g(x) \mod q. \tag{3.8}$$

Il vettore dei suoi coefficienti $\underline{h}$ genera il reticolo NTRU Λ_h, come nel caso del sistema di cifratura NTRUEncrypt.

La chiave privata è data da tali due polinomi uniti ad altri due polinomi $F(x)$, $G(x)$ appartenenti a R tali che verifichino l'uguaglianza

$$f(x) \star G(x) - g(x) \star F(x) = q \tag{3.9}$$

e inoltre

$$\|\underline{F}\|_c \approx \|\underline{f}\|_c \sqrt{\frac{N}{12}} \qquad \|\underline{G}\|_c \approx \|\underline{g}\|_c \sqrt{\frac{N}{12}} \tag{3.10}$$

affinché siano vettori corti e formino una base per il reticolo Λ_h. Una base di Λ_h è data dalle righe di

$$B_h = \begin{bmatrix} \underline{1} & \underline{h} \\ \underline{0} & \underline{q} \end{bmatrix}, \tag{3.11}$$

dove per ogni riga si intende il sottoreticolo generato da quella riga e tutte le sue rotazioni. Il determinante di tale matrice è q.

L'obiettivo è determinare la seconda riga di una matrice avente come prima riga $(\underline{f}, \underline{g})$, tale che anch'essa generi lo stesso reticolo. Dapprima si determinano $(\underline{F}_1, \underline{G}_1)$ tali che la matrice risultante

$$B_1 = \begin{bmatrix} \underline{f} & \underline{g} \\ \underline{F}_1 & \underline{G}_1 \end{bmatrix} \tag{3.12}$$

abbia determinante uguale a 1 modulo $x^N - 1$.

Quando il massimo comun divisore tra due polinomi a coefficienti razionali $u(x)$ e $v(x)$ è 1, esistono altri due polinomi in $\mathbb{Q}[x]$ $a(x)$ e $b(x)$ tali che

$$a(x)u(x) + b(x)v(x) = 1 \tag{3.13}$$

Nel caso in cui i coefficienti di $u(x)$ e $v(x)$ siano interi, non è comunque detto che lo siano anche quelli di $a(x)$ e $b(x)$. Dato che ogni polinomio ha numero finito di coefficienti, si può moltiplicare l'uguaglianza per il minimo

comune multiplo dei denominatori per ottenere un'uguaglianza in $\mathbb{Z}[x]$. Si dice **risultante** di $u(x)$ e $v(x)$ e si indica con $\mathrm{Res}(u, v)$ il più piccolo intero positivo scrivibile come

$$a(x)u(x) + b(x)v(x) = \mathrm{Res}(u, v) \tag{3.14}$$

dove $a(x), b(x) \in \mathbb{Z}[x]$.

Per una trattazione più approfondita di questo argomento, si rimanda a [8].

Si indichino $R_f = \mathrm{Res}(f(x), x^N - 1)$ e $R_g = \mathrm{Res}(g(x), x^N - 1)$, che esistono per la seconda ipotesi fatta su $f(x)$ e $g(x)$. Per (3.14) esistono $u(x)$, $v(x) \in \mathbb{Z}[x]$ tali che

$$f(x) \star v(x) + k_1(x) \star (x^N - 1) = R_f$$
$$g(x) \star u(x) + k_2(x) \star (x^N - 1) = R_g$$

Valutando tali equazioni modulo $x^N - 1$ si ottiene

$$f(x) \star v(x) = R_f \mod (x^N - 1) \tag{3.15}$$
$$g(x) \star u(x) = R_g \mod (x^N - 1) \tag{3.16}$$

L'ipotesi necessaria e sufficiente affinché esistano α e β in $\mathbb{Z}$ che verifichino

$$\alpha R_f + \beta R_g = 1 \tag{3.17}$$

è che $\gcd(R_f, R_g) = 1$. Se questa ipotesi non è verificata, non si può proseguire con questo metodo.

Unendo (3.15), (3.16) a (3.17) si ottiene

$$1 \overset{(3.17)}{=} \alpha R_f + \beta R_g \underset{(3.15)}{\overset{(3.16)}{=}} \alpha(f(x) \star v(x)) + \beta(g(x) \star u(x)) =$$
$$= f(x) \star (\alpha v(x)) - g(x) \star (-\beta u(x)) \tag{3.18}$$

Perciò basta scegliere $G_1(x) = \alpha v(x)$ e $F_1(x) = -\beta u(x)$. La scelta di $F_q(x)$ e $G_q(x)$ affinché la matrice

$$B_q = \begin{bmatrix} f & g \\ F_q & G_q \end{bmatrix} \tag{3.19}$$

abbia determinante q modulo $x^N - 1$ è ora semplice:

$$F_q(x) = qF_1(x) = -q\beta u(x) \qquad G_q(x) = qG_1(x) = q\alpha v(x). \tag{3.20}$$

Infatti

$$\begin{aligned} f(x) \star G_q(x) - g(x) \star F_q(x) &\overset{(3.20)}{=} q[f(x) \star (\alpha v(x)) - g(x) \star (-\beta u(x))] = \\ &\underset{(3.15)}{\overset{(3.16)}{=}} q(\alpha R_f + \beta R_g) \mod x^N - 1 = \\ &\overset{(3.17)}{=} q \mod x^N - 1 \end{aligned}$$

Sfruttando (3.15) e (3.16),

$$f(x)^{-1} = \frac{v(x)}{R_f} \in \frac{\mathbb{Q}[x]}{x^N - 1} \tag{3.21}$$

$$g(x)^{-1} = \frac{u(x)}{R_g} \in \frac{\mathbb{Q}[x]}{x^N - 1} \tag{3.22}$$

$F_q(x)$ e $G_q(x)$ si possono riscrivere mediante tali identità in una maniera che sarà utile in seguito:

$$F_q(x) \overset{(3.20)}{=} -q\beta u(x) \overset{(3.22)}{=} -q\beta R_g g^{-1}(x) \tag{3.23}$$

$$G_q(x) \overset{(3.20)}{=} q\alpha v(x) \overset{(3.21)}{=} q\alpha R_f f^{-1}(x) \tag{3.24}$$

Le righe della matrice M_q generano Λ_h se la matrice di passaggio $B_h B_q^{-1}$ ha determinante 1 e entrate in R. Il primo fatto è vero perché B_h e B_q hanno entrambe determinante q. Il secondo si verifica calcolando esplicitamente la matrice:

$$\begin{aligned} B_h B_q^{-1} = \begin{bmatrix} 1 & \underline{h} \\ 0 & q \end{bmatrix} \begin{bmatrix} \underline{f} & \underline{g} \\ \underline{F}_q & \underline{G}_q \end{bmatrix}^{-1} &= \begin{bmatrix} 1 & \underline{h} \\ 0 & q \end{bmatrix} \begin{bmatrix} \underline{G}_q/q & -\underline{g}/q \\ -\underline{F}_q/q & \underline{f}/q \end{bmatrix}^{-1} = \\ &= \begin{bmatrix} 1 & \underline{H} \\ 0 & q \end{bmatrix} \begin{bmatrix} \underline{G}_1 & -\underline{g}/q \\ -\underline{F}_1 & \underline{f}/q \end{bmatrix}^{-1} = \\ &= \begin{bmatrix} \underline{G}_1 - \underline{F}_1 \star \underline{h} & (-\underline{g} + \underline{f} \star \underline{h})/q \\ -\underline{F}_q & \underline{f} \end{bmatrix}. \end{aligned}$$

Si tratta di polinomi a coefficienti in $\mathbb{Z}$ perché per costruzione $f(x) \star h(x) = g(x) \mod q$.

I polinomi $F_q(x)$ e $G_q(x)$ hanno però coefficienti molto grandi. Per ridurli, si ricorda che, sottraendo a $(\underline{F}_q, \underline{G}_q)$ un multiplo di $(\underline{f}, \underline{g})$, si ottiene ancora una base di Λ_h.

Si desidera dunque minimizzare

$$\|\underline{F}'\|_c = \|\underline{F}_q - \underline{k} \star \underline{f}\|_c$$

al variare di $k(x)$ in R.

La norma è nulla quando $k(x) = F_q(x) \star f^{-1}(x)$. Visto che i coefficienti di $k(x)$ devono essere interi, la scelta ottimale è

$$k(x) = \left\lfloor F_q(x) \star f^{-1}(x) \right\rceil \overset{(3.23)}{=} \left\lfloor (-q\beta R_g)(g^{-1}(x) \star f^{-1}(x)) \right\rceil, \tag{3.25}$$

da cui si ottiene che

$$\begin{aligned} F'(x) &= F_q(x) - k(x) \star f(x) = F_q(x) - \lfloor F_q(x) \star f^{-1}(x) \rceil \star f(x) = \\ &= \left(F_q(x) \star f^{-1}(x) - \lfloor F_q(x) - f^{-1}(x) \rceil\right) \star f(x) = \\ &= \epsilon_f(x) \star f(x) \end{aligned} \tag{3.26}$$

dove $\epsilon_f(x)$ ha coefficienti compresi tra $-\frac{1}{2}$ escluso e $\frac{1}{2}$ incluso, probabilmente equidistribuiti su tale intervallo, perciò

$$\|\underline{\epsilon}_f\|_c \approx \sqrt{\frac{N}{12}}, \tag{3.27}$$

il che porta a concludere che

$$\|\underline{F'}\|_c \approx \|\underline{\epsilon}_f\|_c \|\underline{f}\|_c \approx \|\underline{f}\|_c \sqrt{\frac{N}{12}} \tag{3.28}$$

Affinché si possa ottenere una nuova base per Λ_h, deve essere sottratto a $G_q(x)$ il multiplo $k(x) \star g(x)$. Fortunatamente, si dimostra che il coefficiente ottimale $l(x)$ per minimizzare $\|\underline{G'}\|_c = \|\underline{G}_q - \underline{l} \star \underline{g}\|_c$ e il polinomio $k(x)$ trovato in (3.25) differiscono di poco: ragionando in modo analogo a come si è determinato $k(x)$

$$l(x) = \left\lfloor G_q(x) \star g^{-1}(x) \right\rceil \overset{(3.24)}{=} \left\lfloor (q\alpha R_f)(f^{-1}(x) \star g^{-1}(x)) \right\rceil. \tag{3.29}$$

Confrontando (3.29) e (3.25), si evince che i due fattori ottimali $k(x)$ e $l(x)$ sono arrotondamenti di due polinomi che differiscono soltanto per costante moltiplicativa, quindi è conveniente calcolare dapprima il prodotto di convoluzione tra i due inversi e in seguito moltiplicare per le costanti.

Dividendo (3.9) per $f(x) \star g(x)$ si ottiene che

$$G_q(x) \star g(x)^{-1} - F_q(x) \star f^{-1}(x) = q(f(x)^{-1} \star g(x)^{-1}), \tag{3.30}$$

da cui

$$\begin{aligned} \|\underline{G}_q \star \underline{g}^{-1} - \underline{F}_q \star \underline{f}^{-1}\|_c = \|q(\underline{f}^{-1} \star \underline{g}^{-1})\| &\approx \frac{q}{\|\underline{f}\|_c \|\underline{g}\|_c} = \\ &= \frac{q}{2d\left(1 + \frac{1}{N^2}\right) + \left(1 - \frac{1}{N}\right)^2} \approx \frac{q}{\frac{2}{3}N}. \end{aligned} \tag{3.31}$$

L'ultima uguaglianza segue dal fatto che, come nel capitolo 2, si sceglie $d \approx \frac{N}{3}$ e $\|\underline{f}\|_c = \sqrt{2d\left(1+\frac{1}{N^2}\right)+\left(1-\frac{1}{N}\right)^2} = \|\underline{g}\|_c$. Inoltre, una scelta frequente di q è

$$\frac{1}{3}N \leq q \leq \frac{2}{3}N, \tag{3.32}$$

la quale implica che la norma in (3.31) sia approssimativamente minore o uguale a 1.

Perciò minimizzando la norma di $F'(x)$ si minimizza anche quella del secondo membro.

Con queste premesse, la scelta migliore per minimizzare entrambe le norme è la media arrotondata tra i due coefficienti ottimali $k(x)$ e $l(x)$, cioè

$$p(x) = \left\lfloor \frac{k(x)+l(x)}{2} \right\rceil = \left\lfloor \frac{F_q(x) \star f^{-1}(x) + G_q(x) \star g^{-1}(x)}{2} \right\rceil = \overset{(3.25)}{\underset{(3.29)}{=}} \left\lfloor \frac{-q\beta R_g + q\alpha R_f}{2} \left(f^{-1}(x) \star g^{-1}(x)\right) \right\rceil \tag{3.33}$$

In conclusione, la chiave privata è la matrice

$$B = \begin{bmatrix} \underline{f} & \underline{g} \\ \underline{F} & \underline{G} \end{bmatrix} \tag{3.34}$$

che genera Λ_h, dove

$$F(x) = F_q(x) - p(x) \star f(x) \tag{3.35}$$

$$G(x) = G_q(x) - p(x) \star g(x) \tag{3.36}$$

in cui $p(x)$ viene esplicitato in (3.33), $F_q(x)$ e $G_q(x)$ in (3.23) e (3.24).

Spazio dei message digest

Lo spazio dei message digest $\mathcal{D}$ consiste di tutti i polinomi in R_q, dunque si ha un vettore $\underline{m} = (\underline{m}_1, \underline{m}_2) \in \mathbb{Z}_q^{2N}$. Un'osservazione importante è che se $(\underline{s}, \underline{t}) \in \Lambda_h$ è vicino a $(\underline{m}_1, \underline{m}_2)$, allora, dato $(\underline{x}, \underline{y})$ un punto del reticolo, $(\underline{s}+\underline{x}, \underline{t}+\underline{y})$ è nel reticolo ed è ugualmente vicino a $(\underline{m}_1+\underline{x}, \underline{m}_2+\underline{y})$. Scegliendo $(\underline{x}, \underline{y}) = -(\underline{m}_1, \underline{m}_1 \star \underline{h} \mod q)$, ci si riduce a cercare un elemento del reticolo vicino a $(0, \underline{m}_2 - \underline{m}_1 \star \underline{h})$; perciò, senza perdita di generalità, si considerano in $\mathcal{D}$ solo vettori del tipo $(\underline{0}, \underline{m})$, con $\underline{0}$, $\underline{m} \in \mathbb{Z}_q^N$.

Firma

Alice applica l'algoritmo di Babai con la base privata, ottenendo come firma un vettore del reticolo Λ_h vicino a $(\underline{0}, \underline{m})$.

Si ricavano i coefficienti di $(0, \underline{m})$ rispetto alla base privata:

$$(\underline{\alpha}, \underline{\beta}) = (\underline{0}, \underline{m}) \begin{bmatrix} \underline{f} & \underline{g} \\ \underline{F} & \underline{G} \end{bmatrix}^{-1} = (0, \underline{m}) \begin{bmatrix} \underline{G}/q & -\underline{g}/q \\ -\underline{F}/q & \underline{f}/q \end{bmatrix} = \left(-\frac{\underline{F} \star \underline{m}}{q}, \frac{\underline{f} \star \underline{m}}{q} \right). \tag{3.37}$$

Si arrotondano le entrate all'intero più vicino e, moltiplicando per la basis matrix, si ottiene la firma. In realtà, è sufficiente calcolare le prime N entrate, cioè il vettore $\underline{s}$, perché $\underline{h}$ è pubblica e quindi le restanti entrate si ottengono convolvendo modulo q $\underline{h}$ con $\underline{s}$; questo è vero perché la firma è un elemento del reticolo.

Quindi, il vettore del reticolo vicino a $(0, \underline{m})$ output dell'algoritmo di Babai è $(\underline{s}, \underline{t})$, con

$$\underline{s} = \lfloor (\underline{\alpha}, \underline{\beta}) \rceil \begin{bmatrix} \underline{f} \\ \underline{F} \end{bmatrix}, \tag{3.38}$$

e

$$t = \lfloor (\underline{\alpha}, \underline{\beta}) \rceil \begin{bmatrix} \underline{g} \\ \underline{G} \end{bmatrix} = \underline{s} \star \underline{h} \mod q, \tag{3.39}$$

La firma consiste del solo vettore $\underline{s}$.

Verifica

Per prima cosa Bob calcola mediante l'utilizzo della chiave pubblica l'altra metà del punto del reticolo vicino al documento, nella seconda maniera di (3.39) e l'hash $(0, \underline{m})$ del documento. Infine, si verifica che

$$\| (\underline{s}, \underline{t} - \underline{m}) \mod q \|_c \leq \mathcal{N}, \tag{3.40}$$

cioè che la distanza tra l'hash del documento e la firma sia inferiore al bound. Se ciò è vero, la firma è corretta, altrimenti è falsa. Si nota che

$$\| (\underline{a}, \underline{b}) \mod q \|_c = \min_{\underline{k}_1, \underline{k}_2 \in R} \| (\underline{a} + q\underline{k}_1, \underline{b} + q\underline{k}_2) \|_c$$

Più $\mathcal{N}$ è piccolo, meno è probabile una falsificazione.

Esempio 3.2.2. Si considerino $N = 4$, $q = 61$ e $d = 1$. Siano

$$f(x) = x^3 - x + 1 \qquad g(x) = x^2 - x + 1$$

appartenenti a $\mathcal{T}(2, 1)$, parte della chiave privata.

$f(x)$ è invertibile modulo 61 e

$$f_{61}^{-1}(x) = 12x^3 - 24x^2 - 12x + 25$$

da cui è possibile calcolare la chiave pubblica

$$h(x) = f_{61}^{-1}(x) \star g(x) = 24x^3 + 13x^2 - 25x - 11$$

Alice può rendere nota la chiave pubblica $h(x)$.

Mediante l'algoritmo euclideo si determinano $v(x)$, $u(x) \in R$, e $R_f, R_g \in \mathbb{N}$ tali che

$$\begin{aligned} v(x)f(x) &= R_f \mod x^4 - 1 \\ u(x)g(x) &= R_g \mod x^4 - 1 \end{aligned}$$

ottenendo

$$\begin{aligned} v(x) &= -x^3 + 2x^2 + x + 3 & R_f &= 5 \\ u(x) &= -x^3 + x^2 + 2x + 1 & R_g &= 3 \end{aligned}$$

Il massimo comun divisore tra R_f e R_g è 1, quindi esistono α e β tali che

$$\alpha R_f + \beta R_g = 1,$$

da trovare mediante algoritmo euclideo: $\alpha = -1$, $\beta = 2$.

Alice procede calcolando i due altri polinomi della chiave privata. Dapprima computa:

$$\begin{aligned} F_{61}(x) &= -\beta q u(x) = -122(-x^3 + x^2 + 2x + 1) \\ G_{61}(x) &= \alpha q v(x) = -61(-x^3 + 2x^2 + x + 3) \end{aligned}$$

Gli inversi di f e g in $\mathbb{Q}[x]/(x^4 - 1)$ sono

$$\begin{aligned} f^{-1}(x) &= \frac{1}{R_f} v(x) = \frac{1}{\alpha q R_f} G_{61}(x) = \frac{1}{5}(-x^3 + 2x^2 + x + 3) \\ g^{-1}(x) &= \frac{1}{R_g} u(x) = \frac{1}{-\beta q R_g} F_{61}(x) = \frac{1}{3}(-x^3 + x^2 + 2x + 1) \end{aligned}$$

Il loro prodotto è

$$f^{-1}(x) \star g^{-1}(x) = \frac{1}{15}(x^3 + 8x^2 + 4x + 2)$$

da cui

$$\begin{aligned} p(x) &= \left\lfloor \frac{-q\beta R_g + q\alpha R_f}{2} (f^{-1}(x) \star g^{-1}(x)) \right\rfloor = \\ &= \left\lfloor \frac{61(-2 \cdot 3 + (-1)5)}{2} \frac{1}{15}(x^3 + 8x^2 + 4x + 2) \right\rfloor = \\ &= \left\lfloor -\frac{671}{30}(x^3 + 8x^2 + 4x + 2) \right\rfloor = \\ &= -22x^3 - 179x^2 - 89x - 45 \end{aligned}$$

Il prodotto di tale polinomio con gli elementi già noti della chiave privata sono:

$$p(x) \star f(x) = 112x^3 - 112x^2 - 223x - 112$$
$$p(x) \star g(x) = 68x^3 - 135x^2 - 66x - 202$$

Infine, Alice può calcolare F e G:

$$\begin{aligned} F(x) &= F_{61}(x) - p(x) \star f(x) = \\ &= -122(-x^3 + x^2 + 2x + 1) - (112x^3 - 112x^2 - 223x - 112) = \\ &= 10x^3 - 10x^2 - 21x - 10 \end{aligned}$$

$$\begin{aligned} G(x) &= G_{61}(x) - p(x) \star g(x) = \\ &= -61(-x^3 + 2x^2 + x + 3) - (68x^3 - 135x^2 - 66x - 202) = \\ &= -7x^3 + 13x^2 + 5x + 19 \end{aligned}$$

Alice pubblica infine $\mathcal{N} = 20$.

Bob desidera firmare il message digest $(0, \underline{m})$, con

$$m(x) = 13x^3 - 23x^2 - 12x + 26$$

$$\begin{aligned} (\alpha(x), \beta(x)) &= \left(-\frac{F(x) \star m(x)}{q}, \frac{f(x) \star m(x)}{q} \right) = \\ &= \left(-\frac{733x^3 + 352x^2 - 786x - 423}{61}, \frac{62x^3 + 2x^2 - 61x + 1}{61} \right) \end{aligned}$$

La firma è

$$\begin{aligned} s(x) &= \lfloor (\alpha(x), \beta(x)) \rceil \begin{bmatrix} f(x) \\ F(x) \end{bmatrix} = \\ &= (-12x^3 - 6x^2 - +13x + 6, x^3 - x) \begin{bmatrix} x^3 - x + 1 \\ 10x^3 - 10x^2 - 21x - 10 \end{bmatrix} = \\ &= -31x^2 + x + 31 + 31x^2 - 31 = x \end{aligned}$$

Le restanti entrate del vettore del reticolo vicino al documento sono

$$\begin{aligned} t(x) &= s(x) \star h(x) \mod 61 = \\ &= x \star (24x^3 + 21x^2 - 14x + 7) \mod 61 = \\ &= 21x^3 - 14x^2 + 7x + 24 \mod 61 \end{aligned}$$

La norma della differenza è

$$\begin{aligned}
&\|(s(x), t(x) - m(x) \mod q)\|_c = \\
&\quad = \|(x, 21x^3 - 14x^2 + 7x + 24 - (13x^3 - 23x^2 - 12x + 26) \mod 61)\|_c = \\
&\quad = \sqrt{\|x\|_c^2 + \|8x^3 + 9x^2 + 19x - 2 \mod 61\|_c^2} = \\
&\quad = \sqrt{\frac{3}{4} + 221} \approx 14.89 < 20
\end{aligned}$$

Bob conclude che la firma è corretta perché la norma del vettore è inferiore al bound di Alice.

3.2.1 NTRUSign con perturbazioni

La versione presentata di NTRUSign viene completamente rotta da un'analisi delle trascrizioni, esattamente come la firma GGH. Per scongiurare ciò, si introducono delle perturbazioni: si firma $m' = m + \epsilon$, anziché m: se ϵ è abbastanza piccolo, la firma sarà valida anche per m, ma in questo modo si danno all'attaccante meno informazioni circa la chiave privata. In questo caso, le perturbazioni vengono generate ripetendo il processo di firma in più reticoli segreti simili al reticolo Λ_h.

La costruzione delle chiavi si modifica come segue.

Si generano B reticoli privati $\Lambda_1, \ldots, \Lambda_B$ e un reticolo pubblico Λ_0, nello stesso modo visto per Λ_h. Per ogni reticolo Λ_i si conservano $\{f_i(x), g_i(x), F_i(x), G_i(x), h_i(x)\}$.

La chiave pubblica dello schema consiste unicamente di $h_0(x)$, perciò tutti i reticoli escluso Λ_0 rimangono privati, mentre Λ_0 corrisponde a quello che prima veniva indicato come Λ_h.

La chiave privata consiste degli insiemi $\{f_i(x), g_i(x), F_i(x), G_i(x), h_i(x)\}$, al variare di i in $\{0, \ldots, B\}$, escluso $h_0(x)$.

La procedura di firma consiste nel ripetere per ogni reticolo il metodo descritto sopra, trovando ad ogni step il vettore del reticolo più vicino alla firma ottenuta col precedente, traslata mediante l'elemento del reticolo avente prime entrate uguali a quelle della firma. L'ultimo reticolo che si usa è quello pubblico. Si determina innanzitutto un punto $(\underline{s}_B, \underline{t}_B)$ del reticolo Λ_B vicino a $(0, \underline{m})$, in seguito si trasforma $(\underline{s}_B, \underline{t}_B)$ in un punto $(0, \underline{m}')$, dove

$$\underline{m}' = \underline{t}_B - \underline{s}_B \star \underline{h}_{B-1} \mod q = \underline{s}_B \star (\underline{h}_B - \underline{h}_{B-1}) \mod q,$$

A questo punto si può riapplicare lo stesso procedimento col reticolo Λ_{B-1}, e così via fino a Λ_0. Nelle prime N entrate, la differenza tra il punto dato e il punto del reticolo ad ogni passo è $\underline{s}_i$, perciò la firma finale è

$$\underline{s} = \sum_{i=0}^{B} \underline{s}_i$$

.

Osservazione 3.2.3. È possibile approcciare il problema della riduzione della norma di $F_q(x)$ e $G_q(x)$ in una maniera differente rispetto a quella affrontata sopra: si ottiene un risultato migliore trattando i due polinomi insieme, potendo così operare una generalizzazione dell'algoritmo Round Off di Babai. Il multiplo ottimale di $(f(x), g(x))$ da sottrarre a $(F_q(x), G_q(x))$ che si trova con questo metodo (come indicato in [16]) è

$$k(x) = \left\lfloor \frac{f^\star(x) \star F_q(x) + g^\star(x) \star G_q(x)}{f(x) \star f^\star(x) + g(x) \star g^\star(x)} \right\rceil \tag{3.41}$$

dove con $f^\star(x)$ si intende $f(1/x) \mod x^N - 1$.

Capitolo 4

GPV Framework

Nel 2008 Gentry, Peikert e Vaikuntanathan proposero un nuovo modo di costruire schemi di firma digitali basati sui reticoli, a partire da un algoritmo che, data una base, campiona punti del reticolo distribuiti secondo una distribuzione Gaussiana discreta, avente standard deviation proporzionale alla maggiore lunghezza tra i vettori dell'ortogonalizzazione di Gram-Schmidt della base. In questo capitolo ci si occupa di descrivere tale framework, che è alla base dello schema di firma FALCON.

Definizione 4.0.1. *Dato un reticolo Λ di dimensione n, si definisce il* ***reticolo duale*** *di Λ*

$$\Lambda^* = \{\underline{x} \in \mathbb{R}^n : \langle \underline{x}, \underline{v} \rangle \in \mathbb{Z} \ \forall \, \underline{v} \in \Lambda\} \tag{4.1}$$

Per simmetria, si ha che $(\Lambda^*)^* = \Lambda$.

Data $\mathcal{B} = \{\underline{v}_1, \ldots, \underline{v}_n\}$ una base di Λ, una base di Λ^* si dice base duale ed è un insieme $\mathcal{B}^* = \{\underline{v}_1^*, \ldots, \underline{v}_n^*\}$ tale che

$$\langle \underline{v}_i, \underline{v}_j^* \rangle = \delta_{i,j} \quad \text{per ogni } i, j \text{ in } \{1, \ldots, n\}.$$

4.1 Distribuzione gaussiana discreta

Si tratta di una distribuzione Gaussiana avente supporto discreto, in particolare un reticolo. Fu introdotta da Micciancio e Regev nel 2007 per studiare la complessità computazionale dei problemi sui reticoli. Viene qui introdotta perché lo strumento centrale del GPV framework è una funzione che campiona da questa distribuzione.

Si definisce la **funzione Gaussiana** su $\mathbb{R}^n$ centrata in $\underline{c} \in \mathbb{R}^n$ e di parametro $s > 0$

$$\rho_{s,\underline{c}}(x) = \exp\left(-\frac{\pi \|\underline{x} - \underline{c}\|_2^2}{s^2}\right) \tag{4.2}$$

per ogni $\underline{x}$ appartenente a $\mathbb{R}^n$. Quando omessi, $s = 1$ e $\underline{c} = (0, \ldots, 0)$.

Si può estendere tale funzione a ogni insieme al più numerabile A scrivendo

$$\rho_{s,\underline{c}}(A) = \sum_{\underline{x}\in A} \rho_{s,\underline{c}}(\underline{x}). \tag{4.3}$$

Definizione 4.1.1. *Dati un reticolo Λ di dimensione n, $\underline{c}$ appartenente a $\mathbb{R}^n$ e $s > 0$, si definisce la* ***distribuzione Gaussiana discreta*** *su Λ*

$$D_{\Lambda,s,\underline{c}}(x) = \frac{\rho_{s,\underline{c}}(\underline{x})}{\rho_{s,\underline{c}}(\Lambda)} \tag{4.4}$$

per ogni $\underline{x} \in \mathbb{R}^n$, dove il denominatore è semplicemente un fattore di normalizzazione e si calcola come in (4.3).

Definizione 4.1.2. *Dati un reticolo Λ di dimensione n e un parametro $\epsilon > 0$, si definisce lo* ***smoothing parameter*** *(parametro di lisciezza) di Λ il più piccolo $s > 0$ tale che*

$$\rho_{1/s}(\Lambda^* \setminus \{0\}) \leq \epsilon \tag{4.5}$$

Esso si indica con $\eta_\epsilon(\Lambda)$.

La definizione è ben posta perché

$$\rho_{1/s}(\Lambda^* \setminus \{0\}) = \sum_{\underline{x}\in\Lambda^*\setminus 0} \exp(-\pi s^2 \|\underline{x}\|_2^2)$$

è una funzione decrescente in $s > 0$.

Notazione 4.1.3. Si indica con $\lambda_i(\Lambda)$ il più piccolo raggio tale che la bolla centrata in 0 di tale raggio contenga i vettori linearmente indipendenti del reticolo Λ; quando Λ è chiaro, si può omettere. Quando si considera la norma ℓ_∞, si scrive $\lambda_i^\infty(\Lambda)$.

Data $\mathcal{B}$ una base di un reticolo o B la sua basis matrix, si denota

$$\|\mathcal{B}\| = \|B\| = \max_i \|\underline{b}_i\|_2$$

Indicando con $\{\tilde{\underline{b}}_1, \ldots, \tilde{\underline{b}}_n\}$ l'ortogonalizzazione di Gram-Schmidt della base $\mathcal{B}$, si indica

$$\|B\|_{GS} = \|\mathcal{B}\|_{GS} = \max_i \|\tilde{\underline{b}}_i\|_2$$

Lemma 4.1.4. *Esiste un algoritmo di complessità polinomiale che, data una base $\mathcal{B}$ di un reticolo di dimensione n e un insieme S di n vettori del reticolo linearmente indipendenti ordinati dal più corto al più lungo, dà come output una nuova base del reticolo $\mathcal{T}$ tale che $\|\tilde{t}_i\|_2 \leq \|\tilde{s}_i\|_2$ per ogni $i = 1, \ldots, n$ (intendendo con $\tilde{}$ i vettori dell'ortogonalizzazione di Gram-Schmidt).*

Dimostrazione. [22, pagina 129]. □

S viene dunque efficacemente convertito in una base del reticolo.

Notazione. Una funzione $f(n)$ appartiene a $\omega(g(n))$ se esiste $C > 0$ tale che per ogni $n > N$ $|f(n)| \geq C|g(n)|$.

Lemma 4.1.5. *Per ogni funzione in $\omega(\sqrt{\log n})$, esiste $\epsilon(n) > 0$ trascurabile tale che lo smoothing parameter di un reticolo Λ di dimensione n verifichi*

$$\eta_\epsilon(\Lambda) \leq \frac{\omega(\sqrt{\log n})}{\lambda_1^\infty(\Lambda^*)} \tag{4.6}$$

$$\eta_\epsilon(\Lambda) \leq \omega(\sqrt{\log n}) \cdot \tilde{bl}(\Lambda) \tag{4.7}$$

dove con $\tilde{bl}(\Lambda)$ si indica $\min_{\mathcal{B} \text{ base di } \Lambda} \|\mathcal{B}\|_{GS}$.

Per il lemma 4.1.4, la definizione di $\tilde{bl}(\Lambda)$ considera solo le basi di Λ senza perdita di generalità, perché per ogni insieme S di elementi di Λ di rango massimo esiste una base $\mathcal{T}$ tale che $\|\mathcal{T}\|_{GS} \leq \|S\|_{GS} \leq \|S\|$.

(4.7) si dimostra a partire da (4.6). Sia $\mathcal{B}$ la base tale che $\|\mathcal{B}\|_{GS} = \tilde{bl}(\Lambda)$. Applicando rotazioni rigide e riflessioni al reticolo Λ, che risultano in trasformazioni corrispondenti su Λ^*, si può assumere che l'ortogonalizzazione di Gram-Schmidt di $\mathcal{B}$ abbia vettori paralleli alla base canonica $\{\underline{e}_i\}_i$, cioè

$$\tilde{\underline{b}}_i = \|\tilde{\underline{b}}_i\|_2 \, \underline{e}_i$$

rimane invariato lo smoothing parameter $\eta_\epsilon(\Lambda)$, perché la funzione Gaussiana è invariante rispetto a tali trasformazioni.

Rimane da mostrare che $\lambda_1^\infty(\Lambda^*) \geq 1/\tilde{bl}(\Lambda)$. Sia $\underline{v} \in \Lambda^*$, allora per ogni elemento della base $\underline{b}_i$ si ha che $\langle \underline{v}, \underline{b}_i \rangle \in \mathbb{Z}$. Sia i il più piccolo indice tale che $\langle \underline{v}, \underline{b}_i \rangle$ sia non nullo; tale indice esiste perché la base è formata da vettori linearmente indipendenti e $\underline{v}$ è non nullo. Allora $\underline{v}$ è ortogonale allo spazio generato da $\underline{b}_1, \ldots, \underline{b}_{i-1}$, che per (1.17) è lo stesso generato da $\tilde{\underline{b}}_1, \ldots, \tilde{\underline{b}}_{i-1}$, da cui

$$\begin{aligned} 0 \neq \langle \underline{v}, \underline{b}_i \rangle_2 = \langle \underline{v}, \tilde{\underline{b}}_i + \sum_{j=1}^{i-1} \mu_{i,j} \tilde{\underline{b}}_j \rangle_2 = \langle \underline{v}, \tilde{\underline{b}}_i \rangle_2 = \\ = \|\tilde{\underline{b}}_i\|_2 \langle \underline{v}, \underline{e}_i \rangle_2 = \|\tilde{\underline{b}}_i\|_2 \, \underline{v}_i \end{aligned} \tag{4.8}$$

Il caso $i = 1$ è banale perché $\tilde{\underline{b}}_1 = \underline{b}_1$. Ciò significa che

$$\|\tilde{\underline{b}}_i\|_2 \, |\underline{v}_i| \geq 1,$$

questo implica che

$$\|\underline{v}\|_\infty \geq |v_i| \geq 1/\|\tilde{\underline{b}}_i\| \geq 1/\tilde{bl}(\Lambda) \tag{4.9}$$

Scegliendo $\underline{v}$ la soluzione di SVP su $\Lambda^\star$ rispetto alla norma ∞, si conclude.

Lemma 4.1.6. *Sia Λ un reticolo di dimensione n, $\epsilon \in (0,1)$, $s \geq \eta_\epsilon(\Lambda)$, $\underline{c} \in \mathbb{R}^n$. Allora*

$$\rho_{s,\underline{c}}(\Lambda) \in \left[\frac{1-\epsilon}{1+\epsilon}, 1\right] \cdot \rho_s(\Lambda) \tag{4.10}$$

Ciò significa che la funzione Gaussiana di tutto il reticolo o di un suo traslato è circa la stessa.

Visto che un reticolo Λ è un sottogruppo additivo di $\mathbb{R}^n$, è possibile considerare il quoziente Λ/Λ' per ogni reticolo Λ' contenuto in Λ; esso è un gruppo additivo con elementi $\underline{v} + \Lambda'$, al variare di $\underline{v}$ in Λ.

Corollario 4.1.7. *Dati due reticoli $\Lambda' \subseteq \Lambda$ entrambi di dimensione n, per ogni $\epsilon \in \left(0, \frac{1}{2}\right)$, $s \geq \eta_\epsilon(\Lambda')$ e $\underline{c} \in \mathbb{R}^n$, la distribuzione Gaussiana discreta centrata in $\underline{c}$ di parametro s su Λ modulo Λ' ha distanza statistica al più 2ϵ dall'uniforme in Λ modulo Λ'.*

Ciò significa che un campione Gaussiano su Λ è distribuito circa uniformemente modulo un sottoreticolo Λ', se s è abbastanza grande.

Dimostrazione.

$$\rho_{s,\underline{c}}(\underline{v} + \Lambda') = \rho_{s,\underline{c}-\underline{v}}(\Lambda') \overset{(4.10)}{\in} \left[\frac{1-\epsilon}{1+\epsilon}, 1\right] \rho_{s,\underline{c}}(\Lambda')$$

Per quanto appena visto

$$\begin{aligned}
\rho_{s,\underline{c}}(\Lambda \mod \Lambda') &= \sum_{\underline{v} \in \Lambda \mod \Lambda'} \rho_{s,\underline{c}}(\underline{v} + \Lambda') = \\
&\geq \sum_{\underline{v} \in \Lambda \mod \Lambda'} \rho_{s,\underline{c}}(\Lambda') \cdot \frac{1-\epsilon}{1+\epsilon} = \\
&= \#(\Lambda \mod \Lambda') \cdot \rho_{s,\underline{c}}(\Lambda') \cdot \frac{1-\epsilon}{1+\epsilon}
\end{aligned}$$

Ciò implica che

$$\begin{aligned}
D_{\Lambda,s,\underline{c}}(\underline{v}) \mod \Lambda' &= \frac{\rho_{s,\underline{c}}(\underline{v} \mod \Lambda')}{\rho_{s,\underline{c}}(\Lambda \mod \Lambda')} \leq \\
&\leq \frac{\rho_{s,\underline{c}}(\Lambda')}{\#(\Lambda \mod \Lambda') \cdot \rho_{s,\underline{c}}(\Lambda') \cdot \frac{1-\epsilon}{1+\epsilon}} = \\
&= \frac{1+\epsilon}{1-\epsilon} \cdot \frac{1}{\#(\Lambda \mod \Lambda')}
\end{aligned}$$

La distanza statistica di questa distribuzione dall'uniforme $\mathcal{U}(\Lambda \mod \Lambda')$ è minore o uguale a

$$\frac{1}{2}\sum_{\underline{v}\in\Lambda \mod \Lambda'}\left|\frac{1+\epsilon}{1-\epsilon}\cdot\frac{1}{\#(\Lambda \mod \Lambda')}-\frac{1}{\#(\Lambda \mod \Lambda')}\right| = \\ = \frac{1}{2}\left|\frac{2\epsilon}{1-\epsilon}\right| \overset{\epsilon<1/2}{\leq} 2\epsilon$$

□

Lemma 4.1.8. *Dato un reticolo Λ di dimensione n, $\underline{c}\in\mathbb{R}^n$, $\epsilon\in(0,1)$ e $s\geq\eta_\epsilon(\Lambda)$, $\underline{x}$ distribuito secondo $D_{\Lambda,s,\underline{c}}$ si ha che*

$$\mathbb{P}(\|\underline{x}-\underline{c}\|_2 > s\sqrt{n}) \leq 2^{-n}\frac{1+\epsilon}{1-\epsilon} \tag{4.11}$$

Per più dettagli e dimostrazioni riguardo a questi fatti, si rimanda a [15].

4.2 Campionamento da Gaussiana discreta

L'obiettivo è campionare da una distribuzione Gaussiana discreta $D_{\Lambda,s,\underline{c}}$ a partire da una base $\mathcal{B}$ del reticolo Λ su cui è supportata, quando s è maggiore di $\|\mathcal{B}\|_{GS}$ moltiplicata ad un fattore piccolo. Per prima cosa, si scrive un algoritmo che campiona dal reticolo $\mathbb{Z}$, poi lo si utilizza per scrivere un algoritmo che campiona da un generico reticolo, mediante una variante randomizzata di un algoritmo di Babai detto *Nearest Plane Method*.

4.2.1 Nearest Plane Method

Sia Λ un reticolo di dimensione n con base $\mathcal{B}=\{\underline{b}_1,\dots,\underline{b}_n\}$, avente ortogonalizzazione di Gram-Schmidt $\tilde{\mathcal{B}}=\left\{\underline{\tilde{b}}_1,\dots,\underline{\tilde{b}}_n\right\}$. Questo metodo determina un vettore del reticolo vicino ad un qualsiasi $\underline{w}\in\mathbb{R}^n$, ma funziona in maniera diversa rispetto all'algoritmo Round Off presentato nel capitolo 1, seppur sia sempre dovuto a Babai. Non è detto che l'output di questo algoritmo sia il vettore del reticolo più vicino a $\underline{w}$.

Dato $U=\mathrm{span}(\underline{b}_1,\dots,\underline{b}_{n-1})$, si consideri il sottoreticolo $\Lambda'=\Lambda\cap U$. L'obiettivo è determinare un punto del reticolo $\underline{y}\in\Lambda$ tale che la distanza tra $\underline{w}$ e $U+\underline{y}$ sia minima.

In seguito, si chiama $\underline{w}'$ la proiezione di $\underline{w}$ su $U+\underline{y}$ e $\underline{w}''$ la differenza tra $\underline{w}'$ e $\underline{y}$, appartenente a U.

Si risolve induttivamente il CVP di $\underline{w}''$ in Λ', ottenendo $\underline{y}'\in\Lambda'$. La soluzione al CVP iniziale è dunque $\underline{y}+\underline{y}'$.

Algorithm 3 Algoritmo Nearest Plane di Babai

Input: $\{\underline{b}_1, \ldots, \underline{b}_n\}$ base di Λ, $\underline{w}$ target
Output: $\underline{v}$ soluzione di apprCVP per $\underline{w}$
Determinare $\{\tilde{\underline{b}}_i, \ldots, \tilde{\underline{b}}_n\}$ ortogonalizzazione di GS di $\{\underline{b}_1, \ldots, \underline{b}_n\}$

$\underline{w}_n \leftarrow \underline{w}$
for i=n downto 1 **do**
$\alpha_i \leftarrow \langle \underline{w}_i, \tilde{\underline{b}}_i \rangle_2 / \langle \tilde{\underline{b}}_i, \tilde{\underline{b}}_i \rangle_2$
$\underline{y}_i \leftarrow \lfloor \alpha_i \rceil \underline{b}_i$
$\underline{w}_{i-1} \leftarrow \underline{w}_i - (\alpha_i - \lfloor \alpha_i \rceil)\tilde{\underline{b}}_i - \underline{y}_i$
end for
return $\underline{y} = \underline{y}_1 + \cdots + \underline{y}_n$

Sfruttando (1.17), è possibile scrivere

$$\underline{w} = \sum_{i=1}^{n} \alpha_i \tilde{\underline{b}}_i \qquad \alpha_i \in \mathbb{R} \text{ per ogni } i \tag{4.12}$$

Si desidera determinare $\underline{u} \in U$ tale che la norma

$$\|\underline{w} - (\underline{u} + \underline{y})\|_2^2 \tag{4.13}$$

sia minima, fissato $\underline{y} \in \Lambda$. Scrivendo

$$\underline{u} = \sum_{i=1}^{n-1} \gamma_i \tilde{\underline{b}}_i \qquad \gamma_i \in \mathbb{R} \text{ per ogni } i \tag{4.14}$$

e

$$\underline{y} = \sum_{i=1}^{n} t_i \underline{b}_i = \sum_{i=1}^{n} \tau_i \tilde{\underline{b}}_i \qquad t_i \in \mathbb{Z}, \tau_i \in \mathbb{R} \text{ per ogni } i \tag{4.15}$$

si ha che

$$\begin{aligned} \|\underline{w} - (\underline{u} + \underline{y})\|_2^2 &= \left\| \sum_{i=1}^{n} \alpha_i \tilde{\underline{b}}_i - \left(\sum_{i=1}^{n-1} \gamma_i \tilde{\underline{b}}_i + \sum_{i=1}^{n} \tau_i \tilde{\underline{b}}_i \right) \right\|_2^2 = \\ &= \left\| \sum_{i=1}^{n-1} (\alpha_i - \gamma_i - \tau_i) \tilde{\underline{b}}_i + (\alpha_n - \tau_n) \tilde{\underline{b}}_n \right\|_2^2 = \\ &= \sum_{i=1}^{n-1} (\alpha_i - \gamma_i - \tau_i)^2 \|\tilde{\underline{b}}_i\|_2^2 + (\alpha_n - \tau_n)^2 \|\tilde{\underline{b}}_n\|_2^2 \end{aligned} \tag{4.16}$$

Quindi si sceglie per ogni $i = 1, \ldots, n-1$

$$\gamma_i = \alpha_i - \tau_i. \tag{4.17}$$

In tal modo

$$\|\underline{w} - (\underline{u} + \underline{y})\|_2^2 = (\alpha_n - \tau_n)\|\underline{\tilde{b}_n}\|_2^2 \tag{4.18}$$

La scelta di $\underline{y}$ che minimizza tale norma è quella per cui

$$\tau_n = \lfloor \alpha_n \rceil.$$

Infatti, dato che $\underline{b}_n = \underline{\tilde{b}}_n + \sum_{j=1}^{n-1} \mu_{n,j}\underline{\tilde{b}}_j$, allora $\tau_n = t_n \in \mathbb{Z}$.

Senza perdita di generalità, visto che si considera tutto lo spazio $U + \underline{y}$, si può scegliere

$$\underline{y} = \lfloor \alpha_n \rceil \underline{b}_n \overset{(1.16)}{=} \lfloor \alpha_n \rceil (\underline{\tilde{b}_n} + \sum_{i=1}^{n-1} \mu_{n,i}\underline{\tilde{b}_i}) \tag{4.19}$$

ossia

$$t_i = 0 \qquad \tau_i = \lfloor \alpha_n \rceil \mu_{n,i} \tag{4.20}$$

per ogni i da 1 a $n-1$,

$$t_n = \tau_n = \lfloor \alpha_n \rceil. \tag{4.21}$$

Resta da mostrare che le scelte effettuate sono tali che $\underline{w}' = \underline{u} + \underline{y}$ sia la proiezione di $\underline{w}$ su $U + \underline{y}$, ciò accade se e solo se $\underline{w}'' = \underline{w}' - \underline{y} = \underline{u}$ è la proiezione di $\underline{w} - \underline{y}$ su U.

$$\begin{aligned} \underline{w} - \underline{y} &= \sum_{i=1}^{n} \alpha_i \underline{\tilde{b}}_i - \lfloor \alpha_n \rceil \underline{b}_n = \\ &= \sum_{i=1}^{n-1} (\alpha_i - \lfloor \alpha_i \rceil \mu_{n,i})\underline{\tilde{b}}_i + (\alpha_n - \lfloor \alpha_n \rceil)\underline{\tilde{b}}_n \end{aligned}$$

Dato che per (1.17) $U = \mathrm{span}(\underline{\tilde{b}_1}, \ldots, \underline{\tilde{b}_{n-1}})$,

$$U^{\perp} = \mathrm{span}(\underline{\tilde{b}_n}) \tag{4.22}$$

e perciò

$$P_{U^{\perp}}(\underline{w} - \underline{y}) = (\alpha_n - \lfloor \alpha_n \rceil)\underline{\tilde{b}}_n$$

il che implica che

$$P_U(\underline{w} - \underline{y}) = \underline{w} - \underline{y} - P_{U^{\perp}}(\underline{w} - \underline{y}) = \sum_{i=1}^{n-1} (\alpha_i - \lfloor \alpha_i \rceil \mu_{n,i})\underline{\tilde{b}}_i \overset{(4.17)}{=} \underline{u} = \underline{w}''$$

come volevasi dimostrare. Inoltre

$$\underline{w} = \underline{w} - \underline{y} + \underline{y} = \underline{w}' + (\alpha_n - \lfloor \alpha_n \rceil)\underline{\tilde{b}}_n \tag{4.23}$$

Lemma. *L'output dell'algoritmo $\underline{y}$ appartiene al parallelepipedo*

$$\mathcal{P} = \left\{ \underline{w} + \sum_{i=1}^{n} l_i \tilde{\underline{b}}_i, \text{ dove } -\frac{1}{2} < l_i \leq \frac{1}{2} \text{ per ogni } i \right\} \tag{4.24}$$

Dimostrazione. Questa prova procede per induzione.

$\boxed{n=1}$ Chiaro perché

$$w = \alpha_1 \tilde{\underline{b}}_1 \qquad y_1 = \lfloor \alpha_1 \rceil \underline{b}_1$$

$\tilde{\underline{b}}_1 = \underline{b}_1$ e $\lfloor \alpha_1 \rceil - \alpha_1 \in \left(-\frac{1}{2}, \frac{1}{2}\right]$.

$\boxed{n \geq 2}$ Si applica l'ipotesi induttiva al sottoreticolo Λ':

$$\underline{y}' \in \left\{ \underline{w}'' + \sum_{i=1}^{n-1} l_i \tilde{\underline{b}}_i, \text{ dove } -\frac{1}{2} < l_i \leq \frac{1}{2} \text{ per ogni } i \right\} \tag{4.25}$$

$$\underline{y} = \underline{y}' + \underline{y}_n \in \left\{ \underline{w}'' + \underline{y}_n + \sum_{i=1}^{n-1} l_i \tilde{\underline{b}}_i, \text{ dove } -\frac{1}{2} < l_i \leq \frac{1}{2} \text{ per ogni } i \right\} \tag{4.26}$$

Si ha che

$$\underline{w}'' + \underline{y}_n + \sum_{i=1}^{n-1} l_i \tilde{\underline{b}}_i = \underline{w}' + \sum_{i=1}^{n-1} l_i \tilde{\underline{b}}_i = \underline{w} + \underline{w}' - \underline{w} + \sum_{i=1}^{n-1} l_i \tilde{\underline{b}}_i =$$

$$\overset{(4.23)}{=} \underline{w} + (\lfloor \alpha_n \rceil - \alpha_n) \tilde{\underline{b}}_n + \sum_{i=1}^{n-1} l_i \tilde{\underline{b}}_i \tag{4.27}$$

Si conclude perché $-\frac{1}{2} < \lfloor \alpha_n \rceil - \alpha_n \leq \frac{1}{2}$.

□

Il parallelepipedo $\mathcal{P}$ è un parallelepipedo generato dalla ortogonalizzazione di Gram-Schmidt della base, e ha volume uguale a $\det(\Lambda)$; ciò si vede nella dimostrazione del lemma 1.1.10.

Corollario. *Se $\underline{y}$ è l'output dell'algoritmo, allora*

$$\|\underline{y} - \underline{w}\|^2 \leq \frac{1}{4} \sum_{i=1}^{n} \|\tilde{\underline{b}}_i\|^2 \tag{4.28}$$

Lemma. *Se $\underline{w}$ non appartiene al reticolo, esiste un unico punto del reticolo appartenente a tale parallelepipedo. Se inoltre esiste un elemento $\underline{v} \in \Lambda$ tale che per ogni i nell'insieme $\{1, \ldots, n\}$*

$$\|\underline{v} - \underline{w}\| < \frac{1}{2} \|\tilde{\underline{b}}_i\| \tag{4.29}$$

allora $\underline{v}$ è l'output dell'algoritmo.

Dimostrazione. Si supponga che esistano due punti $\underline{v}$ e $\underline{v}'$ del reticolo in $\mathcal{P}$. Allora

$$\underline{v} = \underline{w} + \sum_{i=1}^{n} l_i \tilde{\underline{b}}_i \qquad \underline{v}' = \underline{w} + \sum_{i=1}^{n} l'_i \tilde{\underline{b}}_i \qquad -\frac{1}{2} < l_i, l'_i \leq \frac{1}{2} \text{ per ogni } i$$

Dato che $\underline{v}$ e $\underline{v}'$ sono entrambi elementi del reticolo lo è anche la loro differenza:

$$\underline{v} - \underline{v}' = \sum_{i=1}^{n} (l_i - l'_i) \tilde{\underline{b}}_i$$

Poiché $(l_n - l'_n)$ è anche il coefficiente di $\underline{b}_n$, esso deve stare in $\mathbb{Z}$; per le restrizioni date l'unica possibilità è che $l_n = l'_n = 0$.

Ora che il coefficiente di $\tilde{\underline{b}}_n$ è nullo, il coefficiente di $\underline{b}_{n-1}$ è lo stesso di $\tilde{\underline{b}}_{n-1}$ e si può replicare il ragionamento ottenendo nuovamente che $l_{n-1} = l'_{n-1} = 0$. Iterando la procedura si ottiene infine che $l_i = l'_i$ per ogni i, cioè che $\underline{v} = \underline{v}'$.

La seconda parte del lemma discende dalla prima perché, per quanto appena dimostrato, un tale $\underline{v}$ deve essere l'unico elemento di $\Lambda \cap \mathcal{P}$. □

Esempio. Sia Λ un reticolo avente basis matrix

$$B = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 0 & -3 \\ 3 & -7 & 3 \end{bmatrix}$$

L'obiettivo è determinare un vettore del reticolo vicino a $\underline{w} = (10, 6, 5)$.

L'intento di questo esempio è confrontare i due metodi ideati da Babai, cioè il Round-Off e il Nearest Plane.

Nearest Plane Si calcola l'ortogonalizzazione di Gram-Schmidt della base, ottenendo

$$\tilde{B} = \begin{bmatrix} \tilde{\underline{b}}_1 \\ \tilde{\underline{b}}_2 \\ \tilde{\underline{b}}_3 \end{bmatrix}$$

dove $\tilde{\underline{b}}_1 = (1, 2, 3)$, $\tilde{\underline{b}}_2 = \left(\frac{24}{7}, \frac{6}{7}, -\frac{12}{7}\right)$ e $\tilde{\underline{b}}_3 = \left(\frac{10}{3}, -\frac{20}{3}, \frac{10}{3}\right)$.

Si scrive $\underline{w}$ in funzione di $\tilde{B}$:

$$\underline{w} = \frac{37}{14} \tilde{\underline{b}}_1 + 2\tilde{\underline{b}}_2 + \frac{3}{20} \tilde{\underline{b}}_3$$

Si procede con l'algoritmo:

$$\alpha_3 = \frac{3}{20} \quad \Longrightarrow \quad \underline{y}_3 = \lfloor \frac{3}{20} \rceil \underline{b}_3 = 0$$

$$\underline{w}_2 = \underline{w} - \frac{3}{20}\tilde{\underline{b}}_3 = \frac{37}{14}\tilde{\underline{b}}_1 + 2\tilde{\underline{b}}_2$$
$$\alpha_2 = 2 \quad \Longrightarrow \quad \underline{y}_2 = 2\underline{b}_2$$
$$\underline{w}_1 = \underline{w}_2 - \underline{y}_2 = \frac{7}{2}\tilde{\underline{b}}_1$$
$$\alpha_1 = \frac{7}{2} \quad \Longrightarrow \quad \underline{y}_1 = 3\tilde{\underline{b}}_1 = 3\underline{b}_1$$

Infine

$$\underline{y} = \underline{y}_1 + \underline{y}_2 + \underline{y}_3 = 3\underline{b}_1 + 2\underline{b}_2 = (9, 6, 3)$$

La distanza tra l'elemento trovato e $\underline{w}$ è

$$\|\underline{w} - \underline{y}\|_2 = \|(10, 6, 5) - (9, 6, 3)\| = \sqrt{5}$$

Round Off L'inversa di B è

$$B^{-1} = \frac{1}{120}\begin{bmatrix} 21 & 27 & 6 \\ 18 & 6 & -12 \\ 21 & -13 & 6 \end{bmatrix}$$

L'output dell'algoritmo è

$$\begin{aligned}
\underline{v} &= \lfloor \underline{w}B^{-1} \rceil B = \\
&= \left\lfloor (10, 6, 5) \cdot \frac{1}{120}\begin{bmatrix} 21 & 27 & 6 \\ 18 & 6 & -12 \\ 21 & -13 & 6 \end{bmatrix} \right\rceil \cdot \begin{bmatrix} 1 & 2 & 3 \\ 3 & 0 & -3 \\ 3 & -7 & 3 \end{bmatrix} = \\
&= \left\lfloor \left(\frac{423}{120}, \frac{241}{120}, \frac{18}{120} \right) \right\rceil \cdot \begin{bmatrix} 1 & 2 & 3 \\ 3 & 0 & -3 \\ 3 & -7 & 3 \end{bmatrix} = \\
&= (4, 2, 0) \cdot \begin{bmatrix} 1 & 2 & 3 \\ 3 & 0 & -3 \\ 3 & -7 & 3 \end{bmatrix} = 4\underline{b}_1 + 2\underline{b}_2 = (10, 8, 6)
\end{aligned}$$

La distanza di $\underline{v}$ da $\underline{w}$ è

$$\|\underline{w} - \underline{v}\|_2 = \|(10, 6, 5) - (10, 8, 6)\|_2 = \sqrt{5}$$

Gli output dei due algoritmi sono differenti, ma soluzioni ugualmente valide del CVP, visto che sono equidistanti dal target.

Osservazione.

$$\underline{w}_j = \underline{w}_{j+1} - (\alpha_{j+1} - \lfloor \alpha_{j+1} \rceil)\tilde{\underline{b}}_{j+1} - \underline{y}_{j+1} = \tag{4.30}$$
$$= \sum_{i=1}^{j} \alpha_i \tilde{\underline{b}}_i + \lfloor \alpha_{j+1} \rceil (\tilde{\underline{b}}_{j+1} - \underline{b}_{j+1}) \tag{4.31}$$

$\underline{w}_j$ si ottiene come la proiezione su span$\{\tilde{\underline{b}}_1, \ldots, \tilde{\underline{b}}_j\}$ di $\underline{w}_{j+1}$ sommata alla differenza tra $\tilde{\underline{b}}_{j+1}$ e $\underline{b}_{j+1}$ moltiplicata per il coefficiente arrotondato. Tale differenza è uguale a

$$\lfloor \alpha_{j+1} \rceil (\tilde{\underline{b}}_{j+1} - \underline{b}_{j+1}) = -\lfloor \alpha_{j+1} \rceil \sum_{i=1}^{j} \mu_{j+1,i} \tilde{\underline{b}}_i$$

Quindi il vettore dei coefficienti rispetto alla GSO di $\underline{w}_j$ è uguale a quello di $\underline{w}_{j+1}$ a cui si sottrae $t_{j+1}\mu_{j+1,:}$.

Ad ogni passo il coefficiente da moltiplicare all'elemento della base $\underline{b}_j$ per ottenere $\underline{y}_j$ è il coefficiente di $\tilde{\underline{b}}_j$ in $\underline{w}_j$ arrotondato all'intero più vicino.

Allora $y_j = t_j \underline{b}_j$, con

$$t_j = \lfloor \alpha_j - \sum_{i=j+1}^{n} t_i \mu_{i,j} \rceil$$

Si osserva inoltre che

$$\underline{\alpha}\tilde{B} = \underline{w} = \underline{s}B = \underline{s}L\tilde{B} \Longrightarrow \underline{\alpha} = \underline{s}L$$

dove L è la matrice avente entrate $(\mu_{i,j})_{i,j}$. Perciò si ottiene che

$$\alpha_j = (\underline{s}L)_j = \sum_{i=1}^{n} s_i L_{i,j} = s_j + \sum_{i=j+1}^{n} s_i L_{i,j}$$

Allora l'iterazione dell'algoritmo Nearest Plane diventa

$$t_j = \lfloor s_j + \sum_{i=j+1}^{n} (s_i - t_i) L_{i,j} \rceil \tag{4.32}$$

4.2.2 Algoritmo di Klein

L'algoritmo di Klein è una variante randomizzata dell'algoritmo Nearest Plane di Babai, sfruttata per campionare da reticoli arbitrari nel framework GPV. Esso fu in principio pensato per risolvere il CVP quando il vettore target è molto vicino al reticolo, ma può essere applicato anche in caso contrario, con maggiori costi computazionali. Ciò che differenzia i due algoritmi è la maniera di arrotondare i coefficienti $\langle \underline{w}_i, \tilde{\underline{b}}_i \rangle_2 / \|\tilde{\underline{b}}_i\|_2^2$, da moltiplicare a $\underline{b}_i$ per determinare $\underline{y}_i$.

La procedura di arrotondamento utilizzata è Sample$\mathbb{Z}$. Un fatto interessante riguardo ad essa è che se l'input è un intero, è possibile che l'output sia un intero diverso.

Campionamento di interi

Per poter definire il campionamento da reticoli arbitrari, è necessario saper campionare la distribuzione $D_{\mathbb{Z},s,c}$: sia $t(n) = \omega(\sqrt{\log n})$, ad esempio $t(n) = \log n$; dati in input s, c, si sceglie un intero random x nell'intervallo $\mathbb{Z} \cap [c - s \cdot t(n), c + s \cdot t(n)]$; l'output è x se $\rho_s(x - c) \in (0, 1]$, altrimenti x viene rigettato e la procedura va ripetuta.

Lemma 4.2.1. *Si consideri il reticolo $\Lambda = \mathbb{Z}$. Per ogni $\epsilon > 0$, $t > 0$, $c \in \mathbb{R}$, $s \geq \eta_\epsilon(\mathbb{Z})$ e $x \sim D_{\mathbb{Z},s,c}$*

$$\mathbb{P}(|x - c| \geq t \cdot s) \leq 2e^{-\pi t^2} \frac{1 + \epsilon}{1 - \epsilon} \tag{4.33}$$

Se $0 < \epsilon < \frac{1}{2}$ e $t \geq \omega(\sqrt{\log n})$, la probabilità che $|x - c| \geq ts$ è trascurabile.

Dimostrazione. Si sfrutta la disuguaglianza di Banaszczyk ([2]) su $(-1, 1)$

$$\rho_s\left((\mathbb{Z} - c) \backslash (t \cdot s \cdot (-1, 1))\right) \leq 2e^{-\pi t^2} \rho_s(\mathbb{Z}) \tag{4.34}$$

La probabilità attribuita da $D_{\mathbb{Z},s,c}$ a tutti gli interi nell'insieme complementare a $t \cdot s \cdot ((-1, 1) + c)$ è, visto che ρ_s è invariante rispetto a traslazioni,

$$D_{\mathbb{Z},s,c}(\mathbb{Z} \setminus (t \cdot s \cdot ((-1, 1) + c))) = \frac{\rho_s(\mathbb{Z} \setminus (t \cdot s \cdot ((-1, 1) + c)))}{\rho_{s,c}(\mathbb{Z})} \leq$$
$$\overset{(4.34)}{\leq} \frac{2e^{-\pi t^2} \cdot \rho_s(\mathbb{Z})}{\rho_{s,c}(\mathbb{Z})} \overset{(4.10)}{\leq} \frac{2e^{-\pi t^2} \cdot \rho_s(\mathbb{Z})}{\frac{1-\epsilon}{1+\epsilon} \cdot \rho_s(\mathbb{Z})}.$$

Questo conclude la prova. □

Lemma 4.2.2. *Per ogni $\epsilon \in (0, e^{-\pi})$, $s \geq \eta_\epsilon(\mathbb{Z})$, $c \in \mathbb{R}$, la distribuzione dell'output di Sample$\mathbb{Z}$ è statisticamente vicina a $D_{\mathbb{Z},s,c}$.*

Dimostrazione. La distribuzione degli output dell'algoritmo *Sample*$\mathbb{Z}$ è proporzionale a $\rho_s(x - c)$ per tutti gli x interi nell'intervallo centrato in c e di ampiezza $2 \cdot s \cdot t(n)$, nulla altrimenti. Il lemma precedente prova che la distribuzione descritta e $D_{\mathbb{Z},s,c}$ sono statisticamente vicine, visto che le due sono pressoché uguali in $[c - s \cdot t, c + s \cdot t]$ e la seconda è molto piccola al di fuori del supporto della prima. □

Campionamento da reticoli arbitrari

Ad ogni iterazione, si sceglie randomicamente un piano $U + \underline{y_i}$ campionando da una Gaussiana discreta sugli interi con certi parametri il coefficiente λ_i tale che $\underline{y_i} = \lambda_i \underline{b_i}$. Questa randomizzazione permette di nascondere la struttura della base segreta agli attaccanti.

Algorithm 4 Algoritmo di Klein

Input: $\{\underline{b}_1, \dots, \underline{b}_n\}$ base di Λ, $\underline{w}$ centro, s>0 std
Output: $\underline{v} \in \Lambda$
Determinare $\{\tilde{\underline{b}}_1, \dots, \tilde{\underline{b}}_n\}$ GSO di $\{\underline{b}_1, \dots, \underline{b}_n\}$

$\underline{w}_n \leftarrow \underline{w}$
for i=n downto 1 **do**
$\alpha_i \leftarrow \langle \underline{w}_i, \tilde{\underline{b}}_i \rangle_2 / \langle \tilde{\underline{b}}_i, \tilde{\underline{b}}_i \rangle_2$
$s'_i \leftarrow s / \|\underline{b}_i\|_2$
$\lambda_i \sim D_{\mathbb{Z}, s'_i, \alpha_i}$
$\underline{y}_i \leftarrow \lambda_i \underline{b}_i$
$\underline{w}_{i-1} \leftarrow \underline{w}_i - \underline{y}_i + (\lambda_i - \alpha_i)\tilde{\underline{b}}_i$
end for
return $\underline{y} = \underline{y}_1 + \cdots + \underline{y}_n$

Si verifica ad ogni passo che $\underline{w}_{i-1}$ appartenga allo spazio generato da $\{\underline{b}_1 \dots, \underline{b}_{i-1}\}$: come nell'algoritmo precedente, ad ogni passo si porta $\underline{w}_i$ su $\text{span}(\tilde{\underline{b}}_1, \dots, \tilde{\underline{b}}_{i-1}) + \underline{y}_i$, e sottraendo $\underline{y}_i$ si ottiene un elemento di $\text{span}(\tilde{\underline{b}}_1, \dots, \tilde{\underline{b}}_{i-1})$. Non è detto che $\underline{y}_i$ sia tale da minimizzare la distanza tra $\underline{w}_i$ e $\text{span}(\tilde{\underline{b}}_1, \dots, \tilde{\underline{b}}_{i-1})$.

Per induzione si prova che l'algoritmo dà come output $\underline{y}$ affinché

$$\underline{y} - \underline{w} = (\lambda_1 - \alpha_1)\tilde{\underline{b}}_1 + \cdots + (\lambda_n - \alpha_n)\tilde{\underline{b}}_n \tag{4.35}$$

Infatti, il caso in cui la dimensione sia 1 è banale perché $\underline{b}_1 = \tilde{\underline{b}}_i$, mentre per il caso generale è sufficiente applicare l'ipotesi induttiva al sottoreticolo Λ' con $\underline{y}' = \underline{y}_1 + \cdots + \underline{y}_{n-1}$ e $\underline{w}_{n-1}$:

$$\underline{y}_1 + \cdots + \underline{y}_{n-1} - w_{n-1} = (\lambda_1 - \alpha_1)\tilde{\underline{b}}_1 + \cdots + (\lambda_{n-1} - \alpha_{n-1})\tilde{\underline{b}}_{n-1}$$

$$\underline{w}_{n-1} = \underline{w} - y_n + (\lambda_n - \alpha_n)\tilde{\underline{b}}_n$$

Sostituendo $\underline{w}_{n-1}$ nell'ipotesi induttiva si ottiene la tesi.

Lemma 4.2.3. *La probabilità che l'algoritmo di campionamento sul reticolo Λ con input $s, \underline{w}$ dia come output $\underline{y} \in \Lambda$ è*

$$\rho_{s,\underline{w}}(\underline{y}) \cdot \prod_{i=1}^{n} \frac{1}{\rho_{s'_i, \alpha_i}(\mathbb{Z})} \tag{4.36}$$

Dimostrazione. Si scriva

$$\underline{y} = \sum_{i=1}^{n} \gamma_i \underline{b}_i \tag{4.37}$$

$\underline{y}$ è l'output della procedura se per ogni i

$$\gamma_i = \lambda_i \tag{4.38}$$

La probabilità che $\gamma_j = \lambda_j$ condizionata all'evento $\gamma_i = \lambda_i$ per ogni $i > j$ è esattamente $D_{\mathbb{Z},s'_i,\alpha_i}(\lambda_i)$. Quindi, usando iterativamente la formula per la probabilità condizionata, si ha che

$$\mathbb{P}(\gamma_1 = \lambda_1, \ldots, \gamma_n = \lambda_n) = \prod_{i=1}^{n} D_{\mathbb{Z},s'_i,\alpha_i}(\lambda_i) = \frac{\prod_{i=1}^{n} \rho_{s'_i,\alpha_i}(\lambda_i)}{\prod_{i=1}^{n} \rho_{s'_i,\alpha_i}(\mathbb{Z})} \tag{4.39}$$

Sfruttando le proprietà della funzione gaussiana e l'ortogonalità dei vettori $\tilde{\underline{b}}_i$, il numeratore si può riscrivere come

$$\begin{aligned}\prod_{i=1}^{n} \rho_{s'_i,\alpha_i}(\lambda_i) &\overset{(4)}{=} \prod_{i=1}^{n} \rho_s((\lambda_i - \alpha_i) \cdot \|\tilde{\underline{b}}_i\|_2) = \\ &= \rho_s\left(\sum_{i=1}^{n} (\lambda_i - \alpha_i)\tilde{\underline{b}}_i\right) = \\ &\overset{(4.35)}{=} \rho_s(\underline{y} - \underline{w}) = \rho_{s,\underline{w}}(\underline{y})\end{aligned}$$

□

Teorema 4.2.4. *Esiste un algoritmo probabilistico che, data una base $\mathcal{B}$ di un reticolo Λ, un parametro $s > \|\mathcal{B}\|_{GS} \cdot \omega(\sqrt{\log n})$ e un centro $\underline{w} \in \mathbb{R}^n$, restituisce come output un campione da una distribuzione statisticamente vicina a $D_{\Lambda,s,\underline{w}}$.*

Dimostrazione. Con tale scelta di s, esiste $\epsilon > 0$ trascurabile in n tale che per ogni i

$$s'_i = \frac{s}{\|\tilde{\underline{b}}_i\|} \geq \omega(\sqrt{\log n}) \overset{(4.7)}{\geq} \eta_\epsilon(\mathbb{Z}) \tag{4.40}$$

visto che $\tilde{bl}(\mathbb{Z}) = 1$, prendendo ad esempio la base canonica. Allora, applicando il lemma 4.2.2, Sample$\mathbb{Z}$ implementa fedelmente l'algoritmo di arrotondamento; inoltre, per (4.10),

$$\rho_{s'_i,\alpha_i}(\mathbb{Z}) \in \left[\frac{1-\epsilon}{1+\epsilon}, 1\right] \rho_{s'_i}(\mathbb{Z}) \tag{4.41}$$

per ogni $\alpha_i \in \mathbb{R}$. Utilizzando il lemma 4.2.3,

$$\begin{aligned}\mathbb{P}(\text{output} = \underline{y}) = \rho_{s,\underline{w}}(\underline{y}) \cdot \prod_{i=1}^{n} \frac{1}{\rho_{s'_i,\alpha_i}(\mathbb{Z})} &\in \left[1, \left(\frac{1+\epsilon}{1-\epsilon}\right)^n\right] \cdot \rho_{s,\underline{w}}(\underline{y}) \cdot \prod_{i=1}^{n} \frac{1}{\rho_{s'_i}(\mathbb{Z})} \\ &\subseteq [1, 1+\epsilon'] \cdot \rho_{s,\underline{w}}(\underline{y}) \cdot \prod_{i=1}^{n} \frac{1}{\rho_{s'_i}(\mathbb{Z})}\end{aligned} \tag{4.42}$$

dove $\epsilon'(n)$ è una funzione trascurabile, e la costante moltiplicata a $[1, 1+\epsilon'] \cdot \rho_{s,\underline{w}}(\underline{y})$ non dipende né da $\underline{w}$ né da $\underline{y}$. Quindi si può sommare al variare di $\underline{y} \in \Lambda$, ottenendo che

$$1 = \sum_{\underline{y} \in \Lambda} \mathbb{P}(\text{output} = \underline{y}) \subseteq [1, 1+\epsilon'] \cdot \sum_{\underline{y} \in \Lambda} \rho_{s,\underline{w}}(\underline{y}) \cdot \prod_{i=1}^{n} \frac{1}{\rho_{s'_i}(\mathbb{Z})} = \tag{4.43}$$

$$= [1, 1+\epsilon'] \cdot \rho_{s,\underline{w}}(\Lambda) \prod_{i=1}^{n} \frac{1}{\rho_{s'_i}(\mathbb{Z})} \tag{4.44}$$

da cui si ricava che

$$\prod_{i=1}^{n} \rho_{s'_i}(\mathbb{Z}) \in [1, 1+\epsilon'] \cdot \rho_{s,\underline{w}}(\Lambda), \tag{4.45}$$

La probabilità di $\underline{y}$ nella distribuzione $D_{\Lambda,s,\underline{w}}$ è

$$\frac{\rho_{s,\underline{w}}(\underline{y})}{\rho_{s,\underline{w}}(\Lambda)} \tag{4.46}$$

quindi la distanza statistica tra le due distribuzioni è

$$\frac{1}{2} \sum_{\underline{y} \in \Lambda} \left| \rho_{s,\underline{w}}(\underline{y}) \cdot \prod_{i=1}^{n} \frac{1}{\rho_{s'_i}(\mathbb{Z})} - \frac{\rho_{s,\underline{w}}(\underline{y})}{\rho_{s,\underline{w}}(\Lambda)} \right| \leq$$

$$\leq \frac{1}{2} \sum_{\underline{y} \in \Lambda} \left| \frac{\rho_{s,\underline{w}}(\underline{y})}{\rho_{s,\underline{w}}(\Lambda)} \cdot \left(\frac{1}{1+\epsilon'} - 1 \right) \right| < \frac{\epsilon'}{2}$$

□

4.3 Trapdoors

Una **trapdoor** (traduzione letterale botola) è una funzione semplice da applicare, ma la cui inversione è difficile, senza conoscere delle informazioni aggiuntive. Si può utilizzare una funzione del genere per creare uno schema di firma: firmare equivale ad applicare l'inversa al messaggio (possibile solo per chi conosce informazioni aggiuntive) e inviare la coppia formata da messaggio e firma, mentre la verifica si attua accertandosi che la funzione applicata alla firma restituisca il messaggio.

4.3.1 Reticoli e distribuzioni GPV

Siano n, m e q degli interi, tipicamente $m = O(n \log n)$ e q un polinomio piccolo in n, ad esempio $q = O(n^3)$. Sia A un elemento di $\mathbb{Z}_q^{m \times n}$; si possono definire a

partire da A due classi di reticoli interi m dimensionali di rango pieno, in cui l'appartenenza di un vettore è determinata unicamente dalle sue entrate modulo q.

$$\Lambda^{\perp}(A) = \{\underline{e} \in \mathbb{Z}^m : \underline{e}A = 0 \mod q\} \tag{4.47}$$

$$\Lambda(A) = \{\underline{y} \in \mathbb{Z}^m : \exists\, \underline{s} \in \mathbb{Z}^n \text{ tale che } \underline{y} = \underline{s}A^T \mod q\} \tag{4.48}$$

Lemma 4.3.1. *Data $A \in \mathbb{Z}_q^{m\times n}$, $\Lambda(A), \Lambda^{\perp}(A)$ definiti come sopra, valgono i seguenti fatti:*

1.

$$\Lambda^{\perp} = q \cdot \Lambda^* \qquad \Lambda = q \cdot (\Lambda^{\perp})^* \tag{4.49}$$

2.

$$\mathbb{Z}^m/\Lambda^{\perp} \cong \{\underline{u} = \underline{e}A \mod q : \underline{e} \in \mathbb{Z}^m\} \subseteq \mathbb{Z}_q^n \tag{4.50}$$

Dimostrazione. 1. Sia $\underline{e} \in \Lambda^{\perp}$, $\underline{y} \in \Lambda$.

$$\langle \underline{e}, \underline{y}\rangle_2 = \langle \underline{e}, \underline{s}A^T\rangle_2 = \langle \underline{e}A, \underline{s}\rangle = 0 \mod q \Longrightarrow \langle \underline{e}, \underline{y}\rangle_2 \in q\mathbb{Z}$$

Ciò implica che $\underline{e} \in q\Lambda^*$. Viceversa, sia $\underline{e} \in q\Lambda^*$, allora $\underline{e} = q\underline{r}$, con $\underline{r} \in \Lambda^*$. Chiaramente,

$$\underline{e}A = q(\underline{r}A) = 0 \mod q$$

Infine,

$$(\Lambda^{\perp})^* = (q \cdot \Lambda^*)^* = \frac{1}{q} \cdot \Lambda$$

2. Si definisca

$$\begin{aligned} \varphi : \mathbb{Z}^m &\to \mathbb{Z}_q^n \\ \underline{e} &\mapsto \underline{e}A \mod q \end{aligned}$$

Utilizzando il primo teorema di isomorfismo

$$\mathbb{Z}^m/\mathrm{Ker}(\varphi) \cong \mathrm{Im}(\varphi)$$

Si conclude osservando che

$$\mathrm{Im}(\varphi) = \{\underline{u} = \underline{e}A \mod q : \underline{e} \in \mathbb{Z}^m\}$$

$$\mathrm{Ker}(\varphi) = \{\underline{e} \in \mathbb{Z}^m : \underline{e}A = 0 \mod q\} = \Lambda^{\perp}$$

□

Lemma 4.3.2. *Sia $A \in \mathbb{Z}_q^{m\times n}$, con $m \geq 2n\log_2 q$. Allora, per tutte le matrici A esclusa una frazione q^n- esima, le somme di sottoinsiemi delle righe di A generano $\mathbb{Z}_q^n$, in altre parole per ogni $\underline{u} \in \mathbb{Z}_q^n$ esiste un $\underline{e} \in \{0,1\}^m$ tale che $\underline{u} = \underline{e}A \mod q$. Più in generale*

$$\{\underline{u} = \underline{e}A \mod q : \underline{e} \in \mathbb{Z}^m\} = \mathbb{Z}_q^n. \tag{4.51}$$

In tal caso, per la seconda parte del lemma 4.3.1, vale il seguente isomorfismo

$$\mathbb{Z}_q^n \cong \mathbb{Z}^m/\Lambda^\perp$$

Lemma 4.3.3. *Assumendo vera (4.51), sia $\epsilon \in \left(0, \frac{1}{2}\right)$ e $s \geq \eta_\epsilon(\Lambda^\perp(A))$.*

Dato $\underline{e} \sim D_{\mathbb{Z}^m,s}$, la distribuzione di $\underline{u} = \underline{e}A \mod q$ ha distanza al più 2ϵ dall'uniforme su $\mathbb{Z}_q^n$.

Inoltre, sia $\underline{t}$ una soluzione arbitraria di $\underline{t}A = \underline{u} \mod q$, dove $\underline{u}$ è un elemento di $\mathbb{Z}_q^n$ fissato; allora, la distribuzione di $\underline{e} \sim D_{\mathbb{Z}^m,s}$ condizionata al fatto che $\underline{e}A = \underline{u} \mod q$, è $\underline{t} + D_{\Lambda^\perp,s,-\underline{t}}$.

Dimostrazione. Per il lemma 4.1.7, dato $\underline{e} \sim D_{\mathbb{Z}^m,s}$, la distribuzione di $\underline{e} \mod \Lambda^\perp$ ha distanza al più 2ϵ dall'uniforme su $\mathbb{Z}^m/\Lambda^\perp$, che è isomorfo a $\mathbb{Z}_q^n$ per il secondo fatto di 4.3.1, mediante la funzione che associa $\underline{e} + \Lambda^\perp$ a $\underline{e}A \mod q$, il che prova la prima parte della tesi.

Si consideri ora $\underline{u} \in \mathbb{Z}_q^n$, e la distribuzione D di $\underline{e} \sim D_{\mathbb{Z}^m,s}$, condizionata al fatto che $\underline{e}A = \underline{u} \mod q$. Il supporto di D è $\underline{t}+\Lambda^\perp$ perché se t è una soluzione ogni altra soluzione si scrive come somma di t e un elemento del nucleo dell'applicazione, che è per l'appunto $\Lambda^\perp$.

$$D(e) = \frac{\rho_s(\underline{e})}{\rho_s(\underline{t}+\Lambda^\perp)} = \frac{\rho_{s,-\underline{t}}(\underline{e}-\underline{t})}{\rho_{s,-\underline{t}}(\Lambda^\perp)} = D_{\Lambda^\perp,s,-\underline{t}}(\underline{e}-\underline{t}) \tag{4.52}$$

Scrivendo $\underline{e} = \underline{t} + \underline{v}$, $\underline{v}$ ha distribuzione $D_{\Lambda^\perp,s,-\underline{t}}$. □

Il lemma seguente prova che, scegliendo una matrice A casualmente, è molto probabile che lo smoothing parameter del reticolo $\Lambda^\perp(A)$ sia piccolo.

Lemma 4.3.4. *Sia q primo, $m \geq 2n\log_2 q$. Allora, per ogni $A \in \mathbb{Z}_q^{m\times n}$ a parte una frazione q^n- esima, si ha*

$$\lambda_1^\infty(\Lambda) \geq \frac{q}{4} \tag{4.53}$$

In particolare, per ogni funzione in $\omega(\sqrt{\log m})$, esiste un $\epsilon(m)$ trascurabile tale che

$$\eta_{\epsilon(m)}(\Lambda^\perp(A)) \leq \omega(\sqrt{\log m}) \tag{4.54}$$

Dimostrazione. Sia C l'insieme di tutti i punti a distanza esattamente $q/4$ dall'origine in $\|\|\|_\infty$. La sua intersezione con $\mathbb{Z}^m$ contiene al più $(q/2)^m$ punti perché C è un quadrato m-dimensionale di lato $q/2$.

Per ogni $\underline{s} \in \mathbb{Z}_q^n$ non nullo, scegliendo A uniformemente su $\mathbb{Z}_q^{m\times n}$, la probabilità che $\underline{s}A^T = \underline{v} \mod q$ per qualche $\underline{v}$ in tale intersezione è al più

$$\frac{(q/2)^m}{q^m} = 2^{-m} \overset{m\geq 2n\log_2 q}{\leq} q^{-2n} \tag{4.55}$$

dove il denominatore è la cardinalità di $\mathbb{Z}_q^m$, perché l'uguaglianza è modulo q. Sommando su tutti i possibili $\underline{s}$ non nulli, si ottiene che tale probabilità è minore o uguale a q^{-n}.

La seconda parte della tesi deriva da:

$$\eta_{\epsilon(m)}(\Lambda^\perp) \overset{(4.6)}{\leq} \frac{\omega(\sqrt{\log m})}{\lambda_1^\infty((\Lambda^\perp)^*)} \overset{4.3.1}{=} \frac{\omega(\sqrt{\log m})}{\lambda_1^\infty(\frac{1}{q}\Lambda)} =$$
$$= \frac{\omega(\sqrt{\log m})}{\frac{1}{q}\lambda_1^\infty(\Lambda)} \leq \frac{\omega(\sqrt{\log m})}{4} = \omega(\sqrt{\log m})$$

□

Teorema 4.3.5. *Siano n un intero positivo e q un primo, sia $m \geq 2n\log_2 q$. Allora, per ogni $A \in \mathbb{Z}_q^{m\times n}$, a parte una frazione $2q^{-n}$-esima, e per ogni $s \geq \omega(\sqrt{\log m})$, la distribuzione di $\underline{u} = \underline{e}A \mod q$, con $\underline{e} \sim D_{\mathbb{Z}^m,s}$, è statisticamente vicina all'uniforme su $\mathbb{Z}_q^n$.*

Dimostrazione. I lemmi 4.3.2 e 4.3.4 insieme danno che, per tutte le A tranne una frazione $2q^{-n}$-esima, le righe di A generano $\mathbb{Z}_q^n$ e $s \geq \eta_\epsilon(\Lambda^\perp(A))$ per qualche funzione trascurabile $\epsilon(m)$. Infine, il lemma 4.3.3 dà la tesi. □

4.3.2 SIS e ISIS

Le trapdoors che verranno costruite verranno considerate sicure supponendo la difficoltà di risoluzione di certi problemi, che qui vengono enunciati.

Definizione 4.3.6. *Siano q un intero, A un matrice in $\mathbb{Z}_q^{m\times n}$ e β un intero positivo. Si dice **SIS** (Small Integer Solution) di parametri q, m, β ($\mathrm{SIS}_{q,m,\beta}$) il problema di determinare una soluzione $\underline{e}$ non nulla in $\mathbb{Z}^m$ di*

$$\begin{cases} \underline{e}A = 0 \mod q \\ \|\underline{e}\|_2 \leq \beta \end{cases} \tag{4.56}$$

*Si dice **ISIS** (Inhomogeneous Small Integer Solution) di parametri q, m, β ($\mathrm{ISIS}_{q,m,\beta}$) il problema di determinare, a partire da $\underline{u} \in \{\underline{t}A : \underline{t} \in \mathbb{Z}^m\} \subseteq \mathbb{Z}_q^n$, una soluzione $\underline{e}$ non nulla in $\mathbb{Z}^m$ di*

$$\begin{cases} \underline{e}A = \underline{u} \mod q \\ \|\underline{e}\|_2 \leq \beta \end{cases} \tag{4.57}$$

Osservazione 4.3.7. 1. SIS è equivalente a trovare un vettore corto in $\Lambda^{\perp}(A)$.

2. Sia $\underline{u} = \underline{t}A$, con $\underline{t} \in \mathbb{Z}^m$. Se la soluzione di ISIS in $\underline{u}$ è $\underline{e}$, si può vedere come vettore errore (ha norma piccola) e si può ottenere

$$\underline{v} = \underline{t} - \underline{e} \in \Lambda^{\perp}, \tag{4.58}$$

visto che $\underline{e}$ e $\underline{t}$ hanno stessa immagine $\underline{u}$.

Quando $\beta \geq \sqrt{m}$ e $m \geq 2n \log q$, con q primo, con alta probabilità si ha che, per il lemma 4.3.2, esiste una soluzione $\underline{e} \in \{0,1\}^m$ di ISIS per ogni $\underline{u} \in \mathbb{Z}_q^n$; infatti,

$$\|\underline{e}\|_2 \leq \sqrt{m} \leq \beta$$

4.3.3 PFSs

Definizione 4.3.8. *Una collezione di **PSFs** (funzioni con preimmagine campionabile) consiste di:*

(a) Un generatore di trapdoors, che dà come output (a, t), dove a descrive il funzionamento della funzione trapdoor f_a con dominio D e codominio R, e t è l'informazione aggiuntiva che permette di invertire facilmente f_a.

(b) Un algoritmo di campionamento sul dominio: campiona $\underline{x}$ da una distribuzione possibilmente non uniforme su D, affinché la sua immagine mediante f_a sia distribuita uniformemente su R.

(c) Un algoritmo di campionamento di preimmagine di un elemento del codominio: sfruttando la conoscenza di t, e dato $\underline{y} \in R$, si campiona $\underline{x} \in D$ dalla distribuzione sul dominio, imponendo che l'immagine mediante f_a debba essere $\underline{y}$.

Non deve esistere un algoritmo di tempo polinomiale che calcoli l'inversa di una delle funzioni della collezione in alcun elemento del codominio, quando non è nota l'informazione aggiuntiva t. Si può anche richiedere la resistenza alle collisioni, quando non è nota t.

Ajtai descrisse in [1] una maniera di costruire una matrice $A \in \mathbb{Z}_q^{m \times n}$, la cui distribuzione sia statisticamente vicina all'uniforme, e un insieme S di m vettori corti e linearmente indipendenti del reticolo $\Lambda^{\perp}(A)$, imponendo che $q = O(n^c)$ per qualche $c > 0$ e che $m \geq 5n \log_2 q$. In particolare, si può ottenere $\|S\| \leq m^{1+\epsilon} = L$ per ogni $\epsilon > 0$ e inoltre, grazie al lemma 4.1.4, si può convertire S in una buona base $\mathcal{T}$ di $\Lambda^{\perp}(A)$, tale che

$$\|\mathcal{T}\|_{GS} \leq \|S\|_{GS} \leq \|S\| \leq L. \tag{4.59}$$

Sia $s \geq L\omega(\sqrt{\log m})$. Si illustra una possibile collezione di PSFs:

(a) Il generatore di trapdoors è la procedura di Ajtai, che fornisce come output $(A, \mathcal{T})$, con $A \in \mathbb{Z}_q^{m\times n}$ e $\mathcal{T}$ buona base di $\Lambda^\perp$.

$$\begin{aligned} f_A : D_n &\to R_n \\ \underline{e} &\mapsto \underline{e}A \mod q \end{aligned}$$

dove

$$D_n = \{\underline{e} \in \mathbb{Z}^m : \|\underline{e}\|_2 \leq s\sqrt{m}\} \qquad R_n = \mathbb{Z}_q^n. \tag{4.60}$$

(b) La distribuzione dell'input è $D_{\mathbb{Z}^m,s}$, che viene campionata mediante l'algoritmo 4.

(c) La procedura di inversione di f_A consiste nel determinare mediante algebra lineare $\underline{t} \in \mathbb{Z}^m$ soluzione di $\underline{t}A = \underline{u} \mod q$ (che esiste con probabilità almeno $1 - q^{-n}$ per il lemma 4.3.2), campionare $\underline{v}$ da $D_{\Lambda^\perp,s,-\underline{t}}$ sfruttando la conoscenza di $\mathcal{T}$ nell'algoritmo 4 e infine fornire $\underline{e} = \underline{t} + \underline{v}$.

Teorema 4.3.9. *La collezione di funzioni descritta sopra è PSFs se il problema* $\mathrm{ISIS}_{q,m,s\sqrt{m}}$ *alla base dell'inversione è computazionalmente difficile. Inoltre, è resistente alle collisioni se* $\mathrm{SIS}_{q,m,2s\sqrt{m}}$ *è difficile.*

Dimostrazione. Per il lemma 4.3.4, esiste una funzione $\epsilon(n)$ trascurabile tale che

$$s \geq L\omega(\sqrt{\log m}) \overset{(4.59)}{\geq} \|\mathcal{T}\|_{GS}\omega(\sqrt{\log m}) \geq \tilde{bl}(\Lambda^\perp)\omega(\sqrt{\log m}) \overset{(4.7)}{\geq} \eta_\epsilon(\Lambda^\perp) \tag{4.61}$$

Perciò si può applicare il lemma 4.1.8 per affermare che, eccetto per probabilità esponenzialmente piccole, un campione $\underline{e}$ della distribuzione $D_{\mathbb{Z}^m,s}$ appartiene a D_n.

Sfruttando (4.61), per tutte le possibili scelte di A tranne una frazione esponenzialmente piccola, $f_A(\underline{e})$ è statisticamente vicina ad essere uniforme su R_n per il teorema 4.3.5.

Utilizzando nuovamente (4.61), il teorema 4.2.4 implica che si possa campionare da una distribuzione statisticamente vicina a $D_{\Lambda^\perp,s,-\underline{t}}$ e il lemma 4.3.3 implica che l'inversione campiona dalla giusta distribuzione condizionata.

Il fatto che queste funzioni siano trapdoors è implicato dal fatto che l'inversione di una di queste equivale alla risoluzione di $\mathrm{ISIS}_{q,m,s\sqrt{m}}$, che si è supposto computazionalmente difficile.

La resistenza alle collisioni è data dalla difficoltà computazionale di $\mathrm{SIS}_{q,m,2s\sqrt{m}}$, perché se vi è una collisione tra $\underline{e}$ ed $\underline{e}'$ distinti, allora

$$\begin{cases} (\underline{e} - \underline{e}')A = 0 \mod q \\ \|\underline{e} - \underline{e}'\|_2 \leq \|\underline{e}\|_2 + \|\underline{e}'\|_2 \leq 2s\sqrt{m} \end{cases}$$

cioè $\underline{e} - \underline{e}'$ è soluzione di SIS. □

È possibile costruire schemi di firma digitale usando la collezione appena descritta: se $(A, \mathcal{T})$ è la coppia fornita dal generatore di trapdoor, A assume il ruolo di chiave pubblica, mentre $\mathcal{T}$ quello di chiave privata. Firmare consiste nell'applicare l'inversa di f_A all'hash del documento. Verificare significa applicare f_A e controllare che sia verificata l'uguaglianza tra hash e quanto ottenuto.

Capitolo 5

Fast Fourier Orthogonalization

5.1 DFT

La trasformata di Fourier discreta permette di rappresentare un vettore complesso mediante una particolare base. Essa ha un costo irrisorio: per un vettore di n elementi sono sufficienti $O(n \log_2 n)$ operazioni, quando n è una potenza di 2. Uno dei suoi utilizzi è velocizzare il prodotto tra polinomi di convoluzione.

Definizione 5.1.1. *La trasformata discreta di Fourier (DFT) è l'operatore lineare*

$$\mathcal{F} : \mathbb{C}^n \to \mathbb{C}^n$$
$$\underline{f} = (f_0, \ldots, f_{n-1}) \mapsto (\widehat{f_0}, \ldots, \widehat{f_{n-1}}) = \widehat{\underline{f}}$$

dove per ogni $k = 0, \ldots, n-1$

$$\widehat{f_k} = \sum_{j=0}^{n-1} f_j \omega^{-jk} \tag{5.1}$$

indicando con ω *la radice n-esima dell'unità* $\exp(\frac{2\pi i}{n})$*, cioè una radice del polinomio* $x^n - 1$*.*

Le radici del polinomio $x^n - 1$ sono tutte e sole della forma ω^j, al variare di j in $\{0, \ldots, n-1\}$. Una radice dell'unità n-esima ξ si dice primitiva se le sue potenze danno tutto l'insieme delle radici dell'unità n-esime, e questo accade solo se $\xi = \omega^k$, dove $\gcd(k, n) = 1$.

Osservazione 5.1.2. Sia $f(x)$ il polinomio avente come coefficienti $\underline{f} = (f_0, \ldots, f_{n-1})$, allora

$$\widehat{f_k} = f(\omega^{-k})$$

da cui si ottiene che

$$\widehat{\underline{f}} = (f(\omega^{-k}))_{k=0,\ldots,n-1} = (f(\omega^k))_{k=n,n-1,\ldots,1}$$

Definizione 5.1.3. *Una* ***matrice di Vandermonde*** *è una matrice le cui righe o colonne hanno elementi, a partire da 1, in progressione geometrica.*

Sia F la matrice avente entrate

$$F_{j,k} = \omega^{-jk} \qquad j,k = 0,\dots,n-1, \tag{5.2}$$

essa ha la forma

$$F = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega^{-1} & \omega^{-2} & \dots & \dots \\ 1 & \omega^{-2} & \omega^{-4} & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ 1 & \omega^{-n+1} & \dots & \dots & \dots \end{bmatrix} \tag{5.3}$$

allora F è una matrice di Vandermonde e inoltre

$$\underline{\widehat{f}} = \underline{f}F \tag{5.4}$$

Lemma 5.1.4. *Sia ξ una radice primitiva n-esima dell'unità, allora*

$$\sum_{j=0}^{n-1} \xi^{jk} = \begin{cases} n & \text{se } k = 0 \mod n \\ 0 & \text{se } k \neq 0 \mod n \end{cases} \tag{5.5}$$

Dimostrazione.

$$x^n - 1 = (x-1)(x^{n-1} + \cdots + x + 1)$$

Visto che $\mathbb{C}$ è un campo, se il primo membro dell'uguaglianza sopra è nullo, uno dei due fattori a secondo membro deve essere nullo.

Quando $k \neq 0 \mod n$, $\xi^k \neq 1$ e dunque, dato che $\xi^{kn} = 1$ per ogni k, vale la tesi.

Quando $k = 0 \mod n$, $\xi^k = 1$ e perciò tale somma è uguale a n. □

Lemma 5.1.5. *La matrice F soddisfa le seguenti proprietà:*

(a) F è simmetrica

(b)

$$F^H F = nI_n, \tag{5.6}$$

dove F^H indica la trasposta coniugata di F.

(c)

$$F^2 = n\Pi_n \tag{5.7}$$

dove Π_n corrisponde alla matrice di permutazione di $\pi = (\pi_j)_j$, con $\pi_j = -j \mod n$.

Dimostrazione. (a) Chiaro perché $\omega^{-jk} = \omega^{-kj}$.

(b) Si considera il prodotto scalare tra la j-esima riga di F^H e la k-esima colonna di F. Dato che F è simmetrica, F^H è semplicemente la matrice avente entrate quelle di F a cui si applica la coniugazione complessa, quindi $F^H = (\omega^{jk})_{j,k=0,\dots,n-1}$, perché $\overline{\omega^{-1}} = \omega$.

$$\begin{aligned}\langle F^H_{j,:}, F_{:,k}\rangle = \langle (\omega^{ji})_i, (\omega^{-ik})_i\rangle = \\ = \sum_{i=0}^{n-1} \omega^{i(j-k)} \overset{(5.5)}{=} \begin{cases} n & \text{se } j = k \mod n \\ 0 & \text{se } j \neq k \mod n \end{cases} = \\ \underset{0\leq j,k<n}{=} n\delta_{j,k}\end{aligned}$$

(c) Si considera il prodotto scalare tra la riga j-esima e la colonna k-esima di F:

$$\begin{aligned}\langle F_{j,:}, F_{:,k}\rangle = \langle (\omega^{-ji})_i, (\omega^{-ik})_i\rangle = \\ = \sum_{i=0}^{n-1} \omega^{-i(j+k)} \overset{(5.5)}{=} \begin{cases} n & \text{se } j+k = 0 \mod n \\ 0 & \text{se } j+k \neq 0 \mod n \end{cases} = \\ = n\delta_{j,-j \mod n}\end{aligned}$$

Se $0 < j < n$, visto che la stessa disuguaglianza deve valere per $-j \mod n$, il rappresentante di tale classe richiesto è $n - j$; se invece $j = 0$, il rappresentante richiesto è 0.

□

Osservazione 5.1.6. Dal lemma precedente si deduce che la trasformata di Fourier è invertibile e la sua inversa è data da

$$F^{-1} \overset{(5.6)}{=} \frac{1}{n} F^H. \tag{5.8}$$

Applicando $\mathcal{F}^{-1}$ ad un vettore $\underline{g}$ si ottiene

$$(\mathcal{F}^{-1}(\underline{g}))_k = \frac{1}{n}\underline{g}F^H = \frac{1}{n}\sum_{j=0}^{n-1} g_j\omega^{jk} = \frac{1}{n}g(\omega^k) \tag{5.9}$$

Osservazione 5.1.7. Seguendo l'approccio dell'osservazione (5.1.2), si considera il polinomio $g(x)$ il cui vettore dei coefficienti è $\underline{g} = (g_0, \dots, g_{n-1})$. Allora

$$\mathcal{F}^{-1}(\underline{g}) = \frac{1}{n}(g(\omega^k))_{k=0,\dots,n-1}$$

Ciò significa che la valutazione di $g(x)$ nelle radici dell'unità è la sua antitrasformata di Fourier, moltiplicata per n.

Lemma 5.1.8. *Siano $\underline{f}, \underline{g}$ appartenenti a $\mathbb{C}^n$. Vale l'uguaglianza di Parseval:*

$$\langle \widehat{\underline{f}}, \widehat{\underline{g}} \rangle_2 = n \langle \underline{f}, \underline{g} \rangle_2 \tag{5.10}$$

Dimostrazione.

$$\langle \widehat{\underline{f}}, \widehat{\underline{g}} \rangle_2 = \langle \underline{f}F, \underline{g}F \rangle_2 = \langle \underline{f}, \underline{g}FF^H \rangle_2 \overset{(5.6)}{=} n \langle \underline{f}, \underline{g} \rangle_2$$

□

Il prodotto tra due polinomi in $R = \frac{\mathbb{Z}[x]}{x^n - 1}$ corrisponde al prodotto di convoluzione tra i rispettivi vettori dei coefficienti. L'obiettivo è mostrare come velocizzare il calcolo usando la trasformata di Fourier.

Dato $\underline{f} = (f_0, \ldots, f_{n-1})$, la traslazione ciclica di lunghezza $p \in \mathbb{Z}$ di $\underline{f}$ si indica con $\underline{f}^p$ e le sue entrate sono $(f^p)_j = f_{(j+p \mod n)}$ al variare di j in $0, \ldots, n-1$.

Lemma 5.1.9.

$$(\widehat{f^p})_k = \widehat{f}_k \omega^{kp} \tag{5.11}$$

Dimostrazione.

$$\begin{aligned}
(\widehat{f^p})_k &= \sum_{j=0}^{n-1} \omega^{-kj} f_j^p = \sum_{j=0}^{n-1} \omega^{-kj} f_{j+p \mod n} = \\
&= \sum_{j=0}^{n-1-p} \omega^{-kj} f_{j+p} + \sum_{j=n-p}^{n-1} \omega^{-kj} f_{j+p-n} = \\
&= \sum_{i=p}^{n-1} \omega^{-k(i-p)} f_i + \sum_{i=0}^{p-1} \underbrace{\omega^{-k(i-p+n)}}_{=\omega^{-k(i-p)}} f_i = \\
&= \sum_{i=0}^{n-1} \omega^{-k(i-p)} f_i = \omega^{kp} \widehat{f}_k
\end{aligned}$$

□

Teorema 5.1.10. *Per ogni $k = 0, \ldots, n-1$*

$$(\widehat{f \star g})_k = (\widehat{f})_k (\widehat{g})_k \tag{5.12}$$

Dimostrazione.

$$\begin{aligned}
(\widehat{f \star g})_k &= \sum_{j=0}^{n-1} (f \star g)_j \omega^{-kj} = \\
&= \sum_{j=0}^{n-1} \sum_{i=0}^{n-1} f_{(j-i \mod n)} g_i \, \omega^{-kj} = \\
&= \sum_{i=0}^{n-1} \left(\sum_{j=0}^{n-1} (f^{-i})_j \omega^{-kj} \right) g_i = \\
&= \sum_{i=0}^{n-1} (\widehat{f^{-i}})_k g_i = \sum_{i=0}^{n-1} (\widehat{f})_k \omega^{-ik} g_i = \\
&= (\widehat{f})_k (\widehat{g})_k
\end{aligned}$$

□

Quindi, per calcolare il prodotto di convoluzione tra due polinomi $f(x), g(x) \in R$, si considerano i vettori dei rispettivi coefficienti $\underline{f}$ e $\underline{g}$, se ne calcola la trasformata di Fourier $\widehat{\underline{f}}$, $\widehat{\underline{g}}$, si moltiplicano questi due vettori componente per componente e al vettore ottenuto si applica la trasformata inversa di Fourier: si ricava il vettore dei coefficienti del prodotto di convoluzione di $f(x)$ e $g(x)$.

5.1.1 FFT

Calcolando la trasformata di Fourier e la sua inversa come prodotto matrice vettore, si compiono n^2 moltiplicazioni e $n^2 - n$ addizioni. Vi è una maniera computazionalmente meno costosa di farlo, dovuta a Cooley e Tukey e detta FFT (Fast Fourier Transform), che ha il costo minore quando n è una potenza di 2.

n pari

Si calcoli ad esempio la trasformata di Fourier inversa di $\underline{g}$.

$$(\mathcal{F}^{-1}(\underline{g}))_k = \frac{1}{n} \sum_{j=0}^{n-1} g_j \omega^{kj}$$

Si divide tale somma in due somme, distinguendo i contributi delle entrate pari e quelli delle entrate dispari:

$$\begin{aligned}
(\mathcal{F}^{-1}(\underline{g}))_k &= \frac{1}{n} \sum_{j=0}^{n/2-1} g_{2j} \omega^{2kj} + \frac{1}{n} \sum_{j=0}^{n/2-1} g_{2j+1} \omega^{k(2j+1)} = \\
&= \frac{1}{n} \sum_{j=0}^{n/2-1} g_{2j} (\omega^2)^{kj} + \frac{1}{n} \omega^k \sum_{j=0}^{n/2-1} g_{2j+1} (\omega^2)^{kj}
\end{aligned}$$

Quando $k < n/2$, il calcolo dell'entrata k-esima si è ridotto a calcolare separatamente la componente k-esima della trasformata inversa del vettore formato dalle entrate pari di $\underline{g}$ e quella del vettore formato dalle entrate dispari, e sommarle moltiplicando la seconda per ω^k: infatti $\omega^2 = \exp\left(\frac{2\pi i}{n}\right)^2 = \exp\left(\frac{2\pi i}{n/2}\right)$.

$$((\mathcal{F}^{-1}(\underline{g}))_0, \dots (\mathcal{F}^{-1}(\underline{g}))_{n/2-1}) = \mathcal{F}^{-1}(\underline{g}_{pari}) + \mathrm{diag}(1, \omega, \dots, \omega^{n/2-1})\mathcal{F}^{-1}(\underline{g}_{dispari}) \tag{5.13}$$

Le ultime $n/2$ entrate di $\mathcal{F}^{-1}(g)$ si calcolano come le prime, sostituendo a k $n/2+k$; ricordando che

$$(\omega^2)^{n/2+i} = (\omega^2)^i \qquad \omega^{n/2+i} = -\omega^i$$

si ricava

$$\begin{aligned}(\mathcal{F}^{-1}(\underline{g}))_{n/2+k} &= \sum_{j=0}^{n/2-1} g_{2j}(\omega^2)^{(k+n/2)j} + \omega^{k+n/2} \sum_{j=0}^{n/2-1} g_{2j+1}(\omega^2)^{(k+n/2)j} = \\ &= \sum_{j=0}^{n/2-1} g_{2j}(\omega^2)^{kj} - \omega^k \sum_{j=0}^{n/2-1} g_{2j+1}(\omega^2)^{kj}\end{aligned}$$

ossia anziché sommare si sottrae.

$$((\mathcal{F}^{-1}(\underline{g}))_{n/2}, \dots (\mathcal{F}^{-1}(\underline{g}))_{n-1}) = \mathcal{F}^{-1}(\underline{g}_{pari}) - \mathrm{diag}(1, \omega, \dots, \omega^{n/2-1})\mathcal{F}^{-1}(\underline{g}_{dispari}) \tag{5.14}$$

Unendo (5.13) e (5.14) e chiamando rispettivamente $\mathcal{F}^{-1}(\underline{g}_{pari}) = \underline{p}$, $\mathcal{F}^{-1}(\underline{g}_{dispari}) = \underline{d}$ e $\mathrm{diag}(1, \omega, \dots, \omega^{n/2-1}) = \Omega$ si ottiene

$$\mathcal{F}^{-1}(\underline{g}) = (\underline{p} + \Omega\underline{d}, \underline{p} - \Omega\underline{d}) \tag{5.15}$$

Questo metodo riduce le operazioni previste al calcolo di 2 trasformate inverse in dimensione $n/2$ ($n^2/4$ moltiplicazioni e $n^2/4 - n/2$ addizioni), $n/2$ moltiplicazioni, $n/2$ addizioni e $n/2$ sottrazioni.

Quando n è una potenza di 2, si può iterare il procedimento con $n/4, n/8$ fino a scomporre totalmente n.

L'ordine finale che viene dato alle entrate del vettore prima di applicare la trasformata alle singole entrate è il cosiddetto **bit reversal order**, cioè ogni indice viene scritto in base 2 (anteponendo zeri quando la lunghezza è minore di $\log_2 n$) e poi l'ordine delle cifre viene invertito.

Esempio. L'obiettivo di questo esempio è chiarire il funzionamento delle due procedure di ordinamento menzionate, e vedere che si equivalgono. Si consideri il vettore $\underline{a} = (a_0, \dots, a_7)$. I passi del primo riordinamento consistono nel separare i vettori ottenuti nelle entrate di indice pari e quelle di indici dispari, e sono i seguenti:

$$((a_0, a_2, a_4, a_6),(a_1, a_3, a_5, a_7))$$
$$((a_0, a_4), (a_2, a_6),(a_1, a_5), (a_3, a_7))$$

L'ultimo passo consiste semplicemente nel separare i vettori di due componenti quindi si ottiene $(a_0, a_4, a_2.a_6, a_1, a_5, a_3, a_7)$.

Nel bit reversal order si scrivono gli indici delle entrate in binario e poi li si legge da destra a sinistra anziché da sinistra a destra, ottenendo il nuovo ordinamento.

$$\begin{aligned}
0 = 000 &\longrightarrow 000 = 0 \\
1 = 001 &\longrightarrow 100 = 4 \\
2 = 010 &\longrightarrow 010 = 2 \\
3 = 011 &\longrightarrow 110 = 6 \\
4 = 100 &\longrightarrow 001 = 1 \\
5 = 101 &\longrightarrow 101 = 5 \\
6 = 110 &\longrightarrow 011 = 3 \\
7 = 111 &\longrightarrow 111 = 7
\end{aligned}$$

L'ordinamento ottenuto è effettivamente lo stesso.

Si dimostra questa corrispondenza in generale per induzione sull'esponente k di 2 nella dimensione del vettore:

- I casi $k = 0$ e $k = 1$ sono banali, il caso $k = 2$ scambia semplicemente l'entrata di indice 1 con quella di indice 2, come fa il bit reversal order (00 e 11 rimangono tali, mentre 10 e 01 si scambiano tra loro).
- Si supponga $k \geq 3$, e la tesi vera fino a $k - 1$. Il vettore di lunghezza 2^k si spezza in due vettori di lunghezza 2^{k-1}, che corrispondono rispettivamente alle entrate pari e a quelle dispari, per cui vale l'ipotesi induttiva; quindi il riordinamento di tali due vettori è il bit reversal order. Si giustappongono i due vettori per determinare l'ordine finale del vettore. Alle rappresentazioni binarie degli indici riordinati di ognuno dei due vettori (per ora identici) si premette uno 0 se fanno parte di quello delle entrate pari e 1 altrimenti, ottenendo la disposizione finale degli indici. Per tornare all'ordinamento originale quello che si fa è nuovamente leggere nella direzione opposta tali rappresentazioni (il bit reversal order applicato due volte è l'identità), quindi le prime 2^{k-1} entrate finiscono per 0, mentre le altre per 1 e le altre cifre si corrispondono quando la posizione nei sottovettori è la stessa; questo prova la tesi.

Costo computazionale Chiamando $c(n)$ la funzione che calcola il costo computazionale della trasformata di Fourier quando n è una potenza di 2, si ha

$$c(n) = 2c\left(\frac{n}{2}\right) + \frac{n}{2}M + nS$$

intendendo con M moltiplicazioni e S addizioni o sottrazioni. Dato che $c(1) = 0$, si mostra induttivamente che

$$c(n) = \left(\frac{n}{2}M + nS\right)\log_2 n = \left(\frac{1}{2}M + S\right) n \log_2 n$$

Caso generale

Si può generalizzare l'idea quando n è composto. Sia il n il prodotto di due interi positivi r_1 e r_2. Nella formula

$$(\mathcal{F}^{-1}(\underline{g}))_k = \sum_{j=0}^{n-1} g_j \omega^{kj}$$

si effettua la divisione con resto per r_1 di j e per r_2 di k:

$$\begin{array}{lll} j = j_1 r_1 + j_0 & 0 \leq j_1 < r_2 & 0 \leq j_0 < r_1 \\ k = k_1 r_2 + k_0 & 0 \leq k_1 < r_1 & 0 \leq k_0 < r_2 \end{array}$$

Da ciò si ottiene che

$$(\mathcal{F}^{-1}(\underline{g}))_k = \sum_{j_0=0}^{r_1-1}\sum_{j_1=0}^{r_2-1} g_{j_0,j_1}\omega^{kj_1r_1}\omega^{kj_0} = \sum_{j_0=0}^{r_1-1}\sum_{j_1=0}^{r_2-1} g_{j_0,j_1}\omega^{k_0j_1r_1}\omega^{(k_1r_2+k_0)j_0}$$

ricordando che $\omega^{kj_1r_1} = \omega^{k_0j_1r_1}$. Il costo si riduce a $n(r_1 + r_2)$ moltiplicazioni e lo stesso numero di addizioni.

5.2 Ortogonalizzazione FFT

Le idee viste nella sezione precedente si possono applicare per velocizzare l'ortogonalizzazione di Gram-Schmidt di una base di un reticolo, quando la sua basis matrix è una matrice a blocchi circolanti aventi dimensione $d \times d$, con d composto. Questo processo a sua volta accelera l'algoritmo Nearest Plane di Babai e le sue varianti, migliorando ad esempio le prestazioni delle trapdoor definite nel GPV framework.

Notazione. In questo capitolo si indica $R_d = \mathbb{R}[x]/(x^d - 1)$ e $Z_d = \mathbb{Z}[x]/(x^d - 1)$.

Alla base di questo algoritmo c'è la rappresentazione di ogni elemento di R_d non più come matrice circolante, ma mediante la torre di anelli

$$\mathbb{R} \subset R_{d_1} \subset R_{d_2} \subset \cdots \subset R_d \tag{5.16}$$

per una certa catena di divisori $1|d_1|d_2|\ldots|d$. Per fare ciò si applica la riindicizzazione vista nell'algoritmo FFT, ossia dividere in ogni vettore le entrate in

base alla classe di equivalenza degli indici modulo il rapporto tra due successivi elementi della catena. Ciò permette di fattorizzare la matrice che, applicata alla basis matrix, restituisce la sua ortogonalizzazione.

Sia $R = \mathbb{R}[x]/(h(x))$, dove $h(x)$ è un polinomio monico con radici distinte su $\mathbb{C}$. Si può dotare R di un'operazione di coniugazione e di un prodotto interno (lineare nel primo fattore, antilineare nel secondo e definito positivo).

Siano $a(x), b(x) \in R$.

Si dice aggiunto di $a(x)$ l'unico elemento $a^\star(x)$ di R tale che per ogni radice ξ di $h(x)$ si ha che $a^\star(\xi) = \overline{a(\xi)}$. Visto che il campo base è $\mathbb{R}$,

$$a^\star(\xi) = \overline{a(\xi)} = a(\bar{\xi})$$

Si definisce un prodotto interno tra $a(x)$ e $b(x)$, o equivalentemente tra i vettori dei coefficienti $\underline{a}$ e $\underline{b}$:

$$\langle a(x), b(x) \rangle = \langle \underline{a}, \underline{b} \rangle = \frac{1}{d} \sum_{h(\xi)=0} a(\xi)\overline{b(\xi)} \tag{5.17}$$

L'aggiunta $B^\star$ di una matrice $B \in R^{m \times n}$ è la trasposta della matrice le cui entrate sono gli aggiunti delle entrate di B. Essa si dice base o di rango massimo se una combinazione lineare delle sue righe è nulla solo se tutti i coefficienti sono nulli.

Nel caso in cui $R = R_d$

$$a^\star(x) = a\left(\frac{1}{x}\right) \mod (x^d - 1) = a_0 + \sum_{i=1}^{d-1} a_i x^{d-i} \tag{5.18}$$

Infatti, visto che tutte le radici dell'unità sono della forma ω^j, con $j = 0, \ldots, n-1$, e che $a(x)$ ha coefficienti reali

$$a^\star(\omega^j) = a(\omega^{-j})$$

che è come valutare il polinomio $a(1/x)$.

Inoltre, il vettore delle valutazioni di $a(x)$ non è altro che la sua antitrasformata di Fourier (o una sua permutazione) moltiplicato per d, quindi

$$\langle a(x), b(x) \rangle = \frac{1}{d} \langle d\mathcal{F}^{-1}(\underline{a}), d\mathcal{F}^{-1}(\underline{b}) \rangle_2 \overset{(5.10)}{=} \langle \underline{a}, \underline{b} \rangle_2 \tag{5.19}$$

5.2.1 Ortogonalizzazione di GS e decomposizione $LDL^\star$

Una matrice $L \in R^{n \times n}$ si dice triangolare inferiore unitaria se è triangolare inferiore e sulla diagonale principale ha tutte entrate uguali a 1.

Una matrice autoaggiunta (uguale alla sua trasposta coniugata) $G \in R^{n \times n}$ si dice di Gram se esiste una matrice $B \in R^{n \times m}$ con $m \geq n$ tale che $G = BB^\star$.

Sia $B \in R^{n\times m}$ una matrice di rango pieno, allora si può decomporre in maniera unica come

$$B = L\tilde{B} \tag{5.20}$$

dove L è triangolare inferiore unitaria e $\tilde{B}$ ha righe perpendicolari a due a due. Si tratta dell'estensione dell'Ortogonalizzazione di Gram-Schmidt (GSO).

Quando R non è più un campo, il procedimento di Gram-Schmidt potrebbe dare dei problemi quando si effettuano le divisioni per la norma dei vettori dell'ortogonalizzazione trovati ai passi precedenti, ma tali norme si mostrano essere sempre invertibili, altrimenti si contraddirebbe che la matrice abbia rango pieno.

Una decomposizione LDL* di una matrice definita positiva è la sua riscrittura nel prodotto $LDL^\star$, con L matrice triangolare inferiore unitaria e D diagonale.

Se G è la matrice di Gram di $B \in R^{n\times m}$ di rango pieno, la decomposizione è data dalla GSO:

$$G = BB^\star = (L\tilde{B})(L\tilde{B})^\star = L(\tilde{B}\tilde{B}^\star)L^\star \tag{5.21}$$

infatti, essendo $\tilde{B}$ con righe a due a due ortogonali, il prodotto con la sua aggiunta è una matrice diagonale con elementi la norma euclidea al quadrato delle valutazioni degli elementi dell'ortogonalizzazione nelle radici del polinomio $h(x)$. L'algoritmo che calcola la decomposizione LDL* di G matrice di Gram di rango pieno si basa totalmente sulla GSO della matrice B che la rende di Gram, da cui è giustificata l'esistenza.

$$(LD)_{i,k} = \sum_{l=1}^{n} L_{i,l}D_{l,k} = L_{i,k}D_{k,k} = \begin{cases} L_{i,k}D_{k,k} & k < i \\ D_{i,i} & k = i \\ 0 & k > i \end{cases}$$

$$G_{i,j} = (LDL^\star)_{i,j} = \sum_{k=1}^{n}(LD)_{i,k}(L^\star)_{k,j} = \sum_{k=1}^{i-1} L_{i,k}D_{k,k}(L_{j,k})^\star + D_{i,i}(L_{j,i})^\star =$$

$$= \begin{cases} \sum_{k=1}^{i-1} L_{i,k}D_{k,k}(L_{j,k})^\star + D_{i,i}(L_{j,i})^\star & j > i \\ \sum_{k=1}^{i-1} L_{i,k}D_{k,k}(L_{i,k})^\star + D_{i,i} & j = i \\ \sum_{k=1}^{j-1} L_{i,k}D_{k,k}(L_{j,k})^\star + L_{i,j}D_{j,j} & j < i \end{cases}$$

L deve essere triangolare inferiore unitaria, quindi

$$L_{i,i} = 1 \qquad L_{i,j} = 0 \quad \forall j > i$$

Dal caso $j < i$ si ricava

$$L_{i,j} = \frac{1}{D_{j,j}}(G_{i,j} - \sum_{k=1}^{j-1} L_{i,k}(L_{j,k})^\star D_{k,k}$$

da quello $j = i$

$$D_{i,i} = G_{i,i} - \sum_{k=1}^{i-1} L_{i,k}(L_{i,k})^\star D_{k,k},$$

Inserendo queste istruzioni all'interno di un ciclo for su $i = 1, \ldots, n$ e le penultime in un ulteriore for per tutti gli indici $j < i$ si ottiene un algoritmo per calcolare le entrate di L e gli elementi della diagonale di D.

Algorithm 5 LDL*

Input: G matrice di Gram
Output: Decomposizione LDL* di G

```
for i = 1, ..., n do
    for j = 1, ..., i − 1 do
        L_{i,j} ← (1/D_{j,j}) (G_{i,j} − Σ_{k=1}^{j−1} L_{i,k}(L_{j,k})^⋆ D_{k,k}
    end for
    D_{i,i} ← G_{i,i} − Σ_{k=1}^{i−1} L_{i,k}(L_{i,k})^⋆ D_{k,k}
end for
return L,D
```

5.2.2 Operatori di linearizzazione

Sia c la mappa che restituisce il vettore dei coefficienti di un polinomio in R_d, estendibile anche ad un vettore o una matrice con entrate in R_d, componente per componente. Si può definire inoltre la mappa C, la quale restituisce la matrice circolante associata ad $a(x) \in R_d$; le due applicazioni sono legate in questo modo:

$$C(a(x)) = \begin{bmatrix} a_0 & a_1 & \ldots & a_{d-1} \\ a_{d-1} & a_0 & \ldots & a_{d-2} \\ \ddots & \ddots & \ddots & \ddots \\ a_1 & a_2 & \ldots & a_0 \end{bmatrix} = \begin{bmatrix} c(a(x)) \\ c(x \star a(x)) \\ \vdots \\ c(x^{d-1} \star a(x)) \end{bmatrix} \in \mathbb{R}^{d \times d} \tag{5.22}$$

Anche C si estende a vettori e matrici componente per componente. Sostanzialmente, c è la rappresentazione del polinomio rispetto alla base $\{1, x, \ldots, x^{d-1}\}$, mentre C è la matrice che si applica per fare il prodotto con $a(x)$.

Vale la seguente:

$$c(a(x))C(b(x)) = c(a(x) \star b(x)) \qquad a(x), b(x) \in R_d, \tag{5.23}$$

tale uguaglianza rende c e C complementari. La prova consiste nell'osservare che:

$$c(a(x))C(b(x)) = c(a(x)) \cdot \begin{bmatrix} c(b(x)) \\ c(x \star b(x)) \\ \vdots \\ c(x^{d-1} \star b(x)) \end{bmatrix} = (a_0, \ldots, a_{d-1}) \cdot \begin{bmatrix} b_0 & b_1 & \ldots & b_{d-1} \\ b_{d-1} & b_0 & \ldots & b_{d-2} \\ \ddots & \ddots & \ddots & \ddots \\ b_1 & b_2 & \ldots & b_0 \end{bmatrix} =$$

$$= \left(\sum_{i+j=k \mod n} a_i b_j \right)_{k=0,\ldots,d-1} = c(a(x) \star b(x))$$

Inoltre, da ciò si deduce che C è moltiplicativa:

$$C(a(x))C(b(x)) = \begin{bmatrix} c(a(x)) \\ c(x \star a(x)) \\ \vdots \\ c(x^{d-1} \star a(x)) \end{bmatrix} \cdot \begin{bmatrix} c(b(x)) \\ c(x \star b(x)) \\ \vdots \\ c(x^{d-1} \star b(x)) \end{bmatrix} =$$

$$\overset{(5.23)}{=} \begin{bmatrix} c(a(x) \star b(x)) \\ c(x \star (a(x) \star b(x))) \\ \vdots \\ c(x^{d-1} ab) \end{bmatrix} = C(a(x) \star b(x))$$

Infine, C commuta con l'operazione di coniugazione:

$$C(a^\star(x)) = C(a(x))^\star \tag{5.24}$$

Allo stesso modo di c e C, si definiscono degli operatori $V_{d/d'}$ e $M_{d/d'}$, dove d' è un elemento della catena di divisori di d. $V_{d/d'}$ rappresenta ogni elemento di R_d come un elemento di $R_{d'}^{d/d'}$, quindi $M_{d/d'}$ rappresenta la matrice di moltiplicazione per l'elemento a cui è applicato in questo nuovo anello.

Sia d il prodotto di primi non necessariamente distinti. Si denota con $\mathrm{gpd}(d)$ il più grande divisore proprio di d. La catena di divisori propri di d che si considera in questa trattazione è sempre quella $1 = d_0|d_1|\ldots|d_{h-1}|d_h = d$, dove per ogni j si ha che $d_{j-1} = gpd(d_j)$; si indica con k_j il rapporto tra d_j e d_{j-1}.

Definizione 5.2.1. *Si consideri $d \in \mathbb{N}^*$, d' appartenente alla torre dei divisori propri e sia $k = d/d'$. Se x è l'indeterminata dell'anello R_d, $y = x^k$ è l'indeterminata dell'anello $R_{d'} = \mathbb{R}[y]/(y^{d'} - 1)$. Si definisce l'operatore $V_{d/d'} : R_d^m \to R_{d'}^{km}$ a seconda dei valori d e d' a partire dalla sua azione sugli elementi in R_d, si estende a R_d^m componente per componente.*

Se $d = 1$, allora $d' = 1$ e l'operatore è l'identità.

Se $d' = \mathrm{gpd}(d)$, si può rappresentare

$$a(x) = \sum_{i \in \mathbb{Z}_k} x^i a_i(y) \qquad a_i(y) \in R_{d'} \quad \forall i \tag{5.25}$$

Allora

$$V_{d/d'}(a(x)) = (a_0(y), \ldots, a_{k-1}(y)) \tag{5.26}$$

Quando d' non è il gpd *di d, $V_{d/d'}$ si definisce ricorsivamente componendo le applicazioni con parametri i diversi elementi della torre tra d e d'. Ad esempio, se $d''|d'|d$*

$$V_{d/d''}(a(x)) = (V_{d'/d''} \circ V_{d/d'})(a(x)) \tag{5.27}$$

V, vista applicata sui vettori dei coefficienti di un polinomio in R_d, agisce come una permutazione dei coefficienti. Ciò che si fa è raggruppare i monomi che abbiano potenza nella stessa classe di resto i modulo k: tali monomi sommati e divisi per x^i costituiscono i polinomi $a_i(y)$, sostituendo $y = x^k$. V opera come il riordino che avviene nella FFT.

Esempio 5.2.2. Sia $d = 8$, $a(x) \in R_8$ tale che

$$\begin{aligned} a(x) &= x + 2x^2 + 3x^3 + 4x^4 + 5x^5 + 6x^6 + 7x^7 = \\ &= (2x^2 + 4x^4 + 6x^6) + x(1 + 3x^2 + 5x^4 + 7x^6) = \\ &= 4x^4 + x(1 + 5x^4) + x^2(2 + 6x^4) + x^3(3 + 7x^4) \end{aligned}$$

Allora, indicando $y = x^2$, $z = x^4$

$$\begin{aligned} V_{8/4}(a(x)) &= (2y + 4y^2 + 6y^3, 1 + 3y + 5y^2 + 7y^3) \\ V_{8/2}(a(x)) &= (V_{4/2} \circ V_{8/4})(a(x)) = (4z, 2 + 6z, 1 + 5z, 3 + 7z) \\ V_{8/1}(a(x)) &= (V_{2/1} \circ V_{4/2} \circ V_{8/4})(a(x)) = (0, 4, 2, 6, 1, 5, 3, 7) \end{aligned}$$

Definizione 5.2.3. *Siano d, d' e k come sopra. Si definisce l'operatore $M_{d/d'} : R_d^{n\times m} \to R_{d'}^{kn\times km}$ a seconda dei valori d e d', a partire dalla sua azione sugli elementi in R_d, si estende a $R_d^{n\times m}$ componente per componente. Se $d = 1$, allora $d' = 1$ e l'operatore è l'identità. Se $d' = \mathrm{gpd}(d)$ e si sfrutta la rappresentazione precedente di $a(x) \in R_d$*

$$M_{d/d'}(a(x)) = \begin{bmatrix} a_0 & a_1 & \ldots & a_{k-1} \\ ya_{k-1} & a_0 & \ldots & a_{k-2} \\ \ddots & \ddots & \ddots & \ddots \\ ya_1 & ya_2 & \ldots & a_0 \end{bmatrix} = \begin{bmatrix} V_{d/d'}(a(x)) \\ V_{d/d'}(x \star a(x)) \\ \vdots \\ V_{d/d'}(x^{k-1} \star a(x)) \end{bmatrix} \tag{5.28}$$

Se d è primo, $M_{d/1}(a(x))$ coincide con $C(a(x))$. Si estende come V nel caso in cui d' non sia il gpd di d.

$M_{d/d'}(a(x))$ rappresenta la matrice di moltiplicazione per $a(x)$ nella stessa base usata da $V_{d/d'}$.

Esempio 5.2.4. Si consideri nuovamente $y = x^2$ e

$$a(x) = 0 + x + 2x^2 + 3x^3 + 4x^4 + 5x^5 + 6x^6 + 7x^7.$$

$$\begin{aligned} M_{8/4}(a(x)) &= \begin{bmatrix} V_{8/4}(a(x)) \\ V_{8/4}(x \star a(x)) \end{bmatrix} = \\ &= \begin{bmatrix} 2y + 4y^2 + 6y^3 & 1 + 3y + 5y^2 + 7y^3 \\ 7 + y + 3y^2 + 5y^3 & 2y + 4y^2 + 6y^3 \end{bmatrix} \end{aligned}$$

Insieme ad altre proprietà, si ottiene la stessa complementarietà di c e C.

Lemma 5.2.5. *Si elencano alcune proprietà degli operatori appena presentati, indicandoli per semplicità M e V:*

1. *M è iniettivo e*

$$M(A \cdot B) = M(A) \cdot M(B) \qquad A, B \in R_d^{n \times m} \tag{5.29}$$

2. *V è una mappa lineare e iniettiva.*

3.
$$V(a(x) \star b(x)) = V(a(x))M(b(x)) \tag{5.30}$$

4. *V è un'isometria:*

$$\langle V(\underline{a}), V(\underline{b}) \rangle_2 = \langle \underline{a}, \underline{b} \rangle_2 \qquad \underline{a}, \underline{b} \in R_d^m \tag{5.31}$$

5. *B è di rango pieno se e solo se lo è $M(B)$ per ogni $B \in R_d^{n \times m}$.*

Dimostrazione.

1. Se $d' = \mathrm{gpd}(d)$, $M_{d/d'}$ è moltiplicativo sugli elementi di R_d per definizione. Usando poi il modo in cui si definisce quando $d''|d'|d$, tale proprietà vale per ogni scelta di parametri. Infine, si sfrutta quanto appena dimostrato insieme al fatto che M è definito componente per componente per arrivare alla tesi con elementi di $R_d^{n \times m}$.

2. Viene direttamente dalla definizione di V.

3. Discende da 1., visto che $V(a(x))$ è la prima riga di $M(a(x))$.

4. Siano $a(x), b(x) \in R_d$,

$$a(x) = \sum_{i \in \mathbb{Z}_{d/\mathrm{gpd}(d)}} x^i a_i(x^{\mathrm{gpd}(d)}) \qquad b(x) = \sum_{i \in \mathbb{Z}_{d/\mathrm{gpd}(d)}} x^i b_i(x^{\mathrm{gpd}(d)})$$

dove

$$a_i(y) = \sum_{j \in \mathbb{Z}_{\mathrm{gpd}(d)}} a_{i,j} y^j \qquad b_i(y) = \sum_{j \in \mathbb{Z}_{\mathrm{gpd}(d)}} b_{i,j} y^j$$

Allora

$$\langle \underline{a}, \underline{b} \rangle_2 = \sum_{i,j} \langle a_{i,j}, b_{i,j} \rangle_2 = \sum_i \langle a_i, b_i \rangle_2 = \langle V(\underline{a}), V(\underline{b}) \rangle_2$$

La tesi si estende a vettori in R_d^m, sfruttando la definizione componente per componente.

5. B ha rango massimo se e solo se $aB = 0 \iff a = 0$. Si può applicare V ad entrambe ottenendo che B ha rango massimo se e solo se $V(aB) = 0 \iff V(a) = 0$. Si sfrutta infine 3., per ottenere che $V(aB) = V(a)M(B)$, il che porta a concludere.

□

Si può applicare V anche al vettore delle valutazioni di $a(x)$ nelle radici d-esime dell'unità che, se ordinate opportunamente, è il vettore dei coefficienti dell'anti-trasformata di Fourier di $a(x)$, moltiplicato per d. In tal caso, per applicare V, si divide per d, si applica la DFT, si permutano i coefficienti e poi si applicano k DFT inverse: si tratta sostanzialmente del procedimento FFT, perciò il costo computazionale è lo stesso: se $a(x) \in R_d$, $V(a(x))$ ha un costo $\Theta(kd)$, $M(a(x))$ $\Theta(k^2 d)$.

5.2.3 Decomposizione LDL* mediante FFT

In questa sottosezione si prova che la matrice L della decomposizione GSO della circolarizzazione di una qualsiasi matrice $B \in R_d^{n \times m}$ si fattorizza come prodotto di matrici sparse e diagonali a blocchi.

Teorema 5.2.6. *Si consideri $d \in \mathbb{N}^*$, e $1 = d_0 | d_1 | \ldots | d_h = d$ la sua torre di divisori propri. Sia $\underline{b} \in R_d^m$ tale che $M_{d/1}(\underline{b})$ sia di rango massimo, allora esiste una GSO di $M_{d/1}(\underline{b})$ tale che:*

$$M_{d/1}(\underline{b}) = \left(\prod_{i=0}^{h-1} M_{d_i/1}(L_i) \right) \cdot \tilde{B}_0 \tag{5.32}$$

in cui $\tilde{B}_0 \in \mathbb{R}^{d \times dm}$ ha righe a due a due ortogonali, e ogni $L_i \in R_{d_i}^{(d/d_i) \times (d/d_i)}$ è una matrice diagonale a blocchi, con d/d_{i+1} blocchi matrici triangolari inferiori unitarie in $R_{d_i}^{k_{i+1} \times k_{i+1}}$.

Dimostrazione. Si prova l'enunciato per induzione sulla lunghezza della torre di divisori di d. Quando d è primo la tesi coincide con la GSO classica. Se invece d è composto, si suppone la tesi vera fino a d_{h-1} nella torre. Per il lemma 5.2.5

$$B_{h-1} = M_{d/d_{h-1}}(\underline{b}) \tag{5.33}$$

è di rango pieno. Si può decomporre B_{h-1} mediante l'ortogonalizzazione di Gram-Schmidt classica, ottenendo

$$B_{h-1} = L_{h-1}\tilde{B} \tag{5.34}$$

con $L_{h-1} \in R_{d_{h-1}}^{k_h \times k_h}$ triangolare inferiore unitaria, $\tilde{B} \in R_{d_{h-1}}^{k_h \times mk_h}$ ortogonale. Le righe di $\tilde{B}$ $\underline{b}_1, \ldots, \underline{b}_{k_h}$ sono ortogonali a due a due e tali che per ogni j $M_{d_{h-1}/1}(\underline{b}_j)$ è di rango massimo. Ad ognuna di queste si può applicare l'ipotesi induttiva:

$$M_{d_{h-1}/1}(\underline{b}_j) = \left(\prod_{i=0}^{h-2} M_{d_i/1}(L_{i,j})\right) \cdot \tilde{B}_j \tag{5.35}$$

con ogni $\tilde{B}_j$ ortogonale e di rango massimo in $\mathbb{R}^{d_{h-1} \times md_{h-1}}$, e $L_{i,j} \in R_{d_i}^{(d_{h-1}/d_i)\times(d_{h-1}/d_i)}$ diagonale a blocchi con d_{h-1}/d_{i+1} blocchi triangolari inferiori unitari in $R_{d_i}^{k_{i+1}\times k_{i+1}}$.

$$\begin{aligned}
M_{d/1}(\underline{b}) = M_{d_{h-1}/1} \circ M_{d/d_{h-1}}(b) &\overset{(5.33)}{=} M_{d_{h-1}/1}(B_{h-1}) = \\
&\overset{(5.34)}{=} M_{d_{h-1}/1}(L_{h-1}) M_{d_{h-1}/1}(\tilde{B}) = \\
&= M_{d_{h-1}/1}(L_{h-1})[M_{d_{h-1}/1}(\underline{b}_1), \ldots, M_{d_{h-1}/1}(\underline{b}_{k_h})] = \\
&\overset{(5.35)}{=} M_{d_{h-1}/1}(L_{h-1}) \cdot \operatorname{diag}\left(\prod_{i=0}^{h-2} M_{d_i/1}(L_{i,j})\right) \cdot [\tilde{B}_1, \ldots, \tilde{B}_{k_h}]
\end{aligned}$$

Siano $L_i = \operatorname{diag}(L_{i,1}, \ldots, L_{i,k_h})$ e $\tilde{B}_0 = [\tilde{B}_1, \ldots, \tilde{B}_{k_h}]$.

Vista la struttura di $L_{i,j}$ per ogni j, L_i è una matrice diagonale a blocchi con $k_h(d_{h-1}/d_{i+1})$ blocchi triangolari inferiori unitari.

Allora

$$\begin{aligned}
M_{d/1}(\underline{b}) &= M_{d_{h-1}/1}(L_{h-1}) \cdot \left(\prod_{i=0}^{h-2} M_{d_i/1}(L_i)\right) \cdot \tilde{B}_0 = \\
&= \left(\prod_{i=0}^{h-1} M_{d_i/1}(L_i)\right) \cdot \tilde{B}_0
\end{aligned}$$

Resta da mostrare che $\tilde{B}_0$ sia ortogonale, ossia che le sue righe siano a due a due perpendicolari.

Se due righe appartengono alla stessa sottomatrice $\tilde{B}_j$, sono già ortogonali.

Ricordando che $\tilde{B}_j$ è l'ortogonalizzazione di $M_{d_{h-1}/1}(\underline{b}_j)$, una sua riga deve essere una combinazione lineare delle righe di $M_{d_{h-1}/1}(\underline{b}_j)$, quindi $\underline{a}_j M_{d_{h-1}/1}(b_j)$. Sia $\alpha_j(x)$ il polinomio avente vettore dei coefficienti $\underline{a}_j$, allora $\underline{a}_j = V(\alpha_j(x))$.

Sia $\underline{a}_k M_{d_{h-1}/1}(\underline{b}_k)$ una riga di $\tilde{B}_k$ e $\underline{a}_l M_{d_{h-1}/1}(\underline{b}_l)$ una riga di $\tilde{B}_l$, con $l \neq k$. Indicando $M_{d_{h-1}/1}$ con M per brevità

$$\begin{aligned}\langle \underline{a}_k M(\underline{b}_k), \underline{a}_l M(\underline{b}_l)\rangle_2 &= \langle V(\alpha_k(x))M(\underline{b}_k), V(\alpha_l(x))M(\underline{b}_l)\rangle_2 = \\ &\overset{(5.30)}{=} \langle V(\alpha_k(x)\star \underline{b}_k), V(\alpha_l(x)\star \underline{b}_l)\rangle_2 = \\ &\overset{(5.31)}{=} \langle \alpha_k(x)\star \underline{b}_k, \alpha_l(x)\star \underline{b}_l\rangle_2 = 0\end{aligned}$$

dove per l'ultima uguaglianza si sfrutta l'ortogonalità tra $\underline{b}_k$ e $\underline{b}_l$. □

Si estende il teorema appena visto da un vettore $\underline{b} \in R_d^m$, ad una matrice $B \in R_d^{n\times m}$.

Corollario 5.2.7. *Sia $d \in \mathbb{N}^*$ e sia $1 = d_0|d_1|\ldots|d_h = d$ la torre di divisori propri di d. Sia B una matrice in $R_d^{n\times m}$ di rango massimo. Allora esistono $h+1$ matrici $L_0, \ldots, L_h$ tali che:*

- *L_h è una matrice triangolare inferiore unitaria in $R_d^{n\times n}$.*
- *Per ogni $i < h$, L_i è una matrice diagonale a blocchi in $R_{d_i}^{n(d/d_i)\times n(d/d_i)}$ i cui $n(d/d_{i+1})$ blocchi sono triangolari inferiori unitarie in $R_{d_i}^{k_{i+1}\times k_{i+1}}$.*

La GSO di $M_{d/1}(B)$ è

$$L \cdot \tilde{B}_0,$$

dove

$$L = \prod_{i=0}^{h} M_{d_i/1}(L_i) \qquad \tilde{B}_0 = L^{-1} M_{d/1}(B)$$

Infine, la decomposizione LDL* *di $M_{d/1}(BB^\star)$ è data da*

$$L \cdot (\tilde{B}_0 \tilde{B}_0^T) \cdot L^T$$

Dimostrazione. Mediante l'ortogonalizzazione di Gram-Schmidt

$$B = L_h B'$$

con L_h triangolare inferiore unitaria e B' ortogonale. Si può applicare il teorema precedente a ogni riga di B' $\underline{b}'_1, \ldots, \underline{b}'_n$: si ottengono $(\tilde{B}_j)_j$ matrici ortogonali e $(L_{i,j})_{i,j}$ diagonali a blocchi. Scegliendo $L_i = \mathrm{diag}(L_{i,j})$ e $\tilde{B}_0 = [\tilde{B}_1, \ldots, \tilde{B}_n]$ si ottiene la GSO, da cui si ottiene anche la decomposizione LDL*. □

Osservazione 5.2.8. Si osserva che ogni matrice a blocchi L_i si può scrivere come prodotto di matrici sparse, che si possono vedere a blocchi, con un unico blocco non nullo.

$$L_i = \text{diag}(L_{i,1}, \dots, L_{i,k}) = \text{diag}(L_{i,1}, 0, \dots, 0) \cdot \text{diag}(0, L_{i,2}, 0, \dots, 0) \cdot \text{diag}(0, \dots, 0, L_{i,k})$$

$$L = \prod_{i=0}^{h} M_{d_i/1}(L_i) = \prod_{i=0}^{h} \prod_{j=1}^{k} M_{d_i/1}(\text{diag}(0, \dots, \underbrace{L_{i,j}}_{\text{posiz } j+1}, \dots, 0)) =$$

$$= \prod_{i=0}^{h} \prod_{j=1}^{k} \text{diag}(0, \dots, M_{d_i/1}(L_{i,j}), \dots, 0)$$

Usando sempre la torre di divisori in cui ogni elemento è il gpd del successivo, si può scrivere un algoritmo per il calcolo veloce della matrice L.

Algorithm 6 Algoritmo di decomposizione LDL* ffLDL$^\star_{R_d}$

Input: $G \in R_d^{n \times n}$ matrice di Gram
Output: Decomposizione compatta $LDL^\star$

```
if d = 1 then
return (L, D) = LDL*(G) (mediante 5)
end if
d' ← gpd(d)
for i=1, ..., n do
    L_i ← ffLDL*_{R_d'}(M_{d/d'}(D_{i,i}))
end for
return (L, (L_i)_i)
```

Albero LDL*

Si rappresenta la fattorizzazione di L mediante un albero, i cui nodi sono matrici strutturate di varie dimensioni. Si sceglie di implementare un algoritmo per la decomposizione LDL* anziché per la GSO, perché nel caso in cui verrà utilizzato ridurrà di molto la complessità computazionale, visto che $n = 2$.

La radice dell'albero è il blocco non banale di una matrice a blocchi triangolare inferiore unitaria. Applicando l'operatore $M_{d_{h-1}/1}$ a tale matrice si ottiene la prima matrice della fattorizzazione di L.

Lo stesso vale per i nodi del livello l-esimo: ogni nodo è il blocco non banale di una matrice triangolare inferiore unitaria, a cui si applica $M_{d_{h-1-l}/1}$. Infine, le matrici ottenute diventano i blocchi diagonali di una matrice diagonale a blocchi, fattore di L.

La procedura utilizzata è ricorsiva, e chiama l'algoritmo LDL* semplice quando $d = 1$, ossia nel caso base. Ad ogni chiamata determina una delle matrici diagonali a blocchi L_i, insieme alla sua rappresentazione mediante albero. Parte dai nodi superiori e via via arriva in fondo.

5.2.4 Fast Fourier Nearest Plane

Si consideri ora $Z_d = \mathbb{Z}[x]/(x^d - 1)$. Il primo algoritmo si deriva da (4.32).

Algorithm 7 Algoritmo Nearest Plane $\mathrm{NP}_{\mathbb{R}}$

Input: $\underline{t} \in \mathbb{R}^n$, decomposizione GSO di $B = L\tilde{B}$
Output: $\underline{z} \in \mathbb{Z}^n$ tale che $(\underline{t} - \underline{z})B \in \mathcal{P}(\tilde{B})$

$\underline{z} \leftarrow 0$
for $j = n, \ldots, 1$ **do**
  $\bar{t}_j \leftarrow t_j + \sum_{i>j}(t_i - z_i)L_{i,j}$
  $z_j \leftarrow \lfloor \bar{t}_j \rceil$
end for
return $\underline{z}$

Algorithm 8 Algoritmo Fast Fourier Nearest Plane $ffNP_{R_d}(t, \mathcal{L})$

Input: $\underline{t} \in R_d^n$, $B \in R_d^{n \times m}$, un albero $\mathcal{L}$ di decomposizione $LDL^\star$ compatta della matrice di Gram di B
Output: $\underline{z} \in Z_d^n$ tale che $V((\underline{z} - \underline{z})B) \in \mathscr{F}_{\tilde{B}_0}$, dove $\tilde{B}_0$ è l'ortogonalizzazione di $M(B)$

if $d = 1$ **then**
  $(L, D) \leftarrow \mathcal{L}$
  return $\mathrm{NP}_{\mathbb{R}}(L, t)$ (mediante 7)
end if
$(L, (\mathcal{L}_i)_i) \leftarrow \mathcal{L}$
$d' \leftarrow \mathrm{gpd}(d)$
for j=n, ..., 1 **do**
  $\bar{t_j} \leftarrow t_j + \sum_{i>j}(t_i - x_i)L_{i,j}$
  $z_j \leftarrow V_{d/d'}^{-1}[ffNP_{R_{d'}}(V_{d/d'}(\bar{t_j}), \mathcal{L}_j))]$
end for
return $z = (z_1, \ldots, z_n)$

Lemma 5.2.9. *Sia $B = \{\underline{b}_1, \ldots, \underline{b}_n\} \in R_d^{n \times m}$ e sia $\tilde{B} = \{\tilde{\underline{b}_1}, \ldots, \tilde{\underline{b}_n}\}$ la sua GSO in R_d. I vettori $\underline{\bar{t}} = (\bar{t_1}, \ldots, \bar{t_n})$ e $\underline{z}$ ottenuti dall'algoritmo 8 verificano la seguente uguaglianza:*

$$(\underline{z} - \underline{t}) \cdot B = (\underline{z} - \underline{\bar{t}}) \cdot \tilde{B} \tag{5.36}$$

Dimostrazione. Ricordando che:

(1) $\bar{t_j} = t_j + \sum_{i<j}(t_i - x_i)L_{i,j}$

(2) $L_{i,i} = 1$

(3) $\tilde{b_j} = b_j - \sum_{i>j} L_{j,i}\tilde{b_i}$

$$
\begin{aligned}
(\underline{z} - \underline{\bar{t}}) \cdot \tilde{B} &= \sum_{j=1}^{n}(z_j - \bar{t_j})\tilde{b_j} = \\
&\overset{(1)}{=} \sum_{j=1}^{n}[(z_j - t_j) + \sum_{i>j} L_{i,j}(z_i - t_i)]\tilde{b_j} = \\
&\overset{(2)}{=} \sum_{j=1}^{n}\sum_{i=j}^{n}(z_i - t_i)L_{i,j}\tilde{b_j} = \\
&= \sum_{i=1}^{n}\left[(z_i - t_i)\sum_{j=1}^{i} L_{i,j}\tilde{b_j}\right] = \\
&\overset{(3)}{=} \sum_{i=1}^{n}(z_i - t_i)b_i = (\underline{z} - \underline{t}) \cdot B
\end{aligned}
$$

□

Teorema 5.2.10. *L'output $\underline{z} \in R_d^n$ dell'algoritmo 8 è tale che*

$$V_{d/1}((\underline{z} - \underline{t})B) \in \mathscr{F}_{\tilde{B_0}} \tag{5.37}$$

dove $\tilde{B_0}$ è l'ortogonalizzazione di $M_{d/1}(B)$ su $\mathbb{R}$.

Dimostrazione. Per $d = 1$ è già stato dimostrato. Per definizione, ogni sottoalbero $\mathscr{L}_j$ è la decomposizione compatta $LDL^\star$ di $M_{d/\mathrm{gpd}(d)}(\tilde{b_j})$. Per ipotesi induttiva, si ha che $V((z_j - \bar{t_j})\tilde{b_j}) \in \mathscr{F}_{\tilde{B}_j}$, in cui $\tilde{B}_j$ è l'ortogonalizzazione di $B_j = M_{d/1}(\underline{\tilde{b}_j})$ Per (5.36) , si ha che

$$(\underline{z} - \underline{t}) \cdot B = \sum_{j=1}^{n}(z_j - \bar{t_j})\tilde{b_j}$$

il che implica che $V((\underline{z} - \underline{t})B) \in \mathscr{F}_{\tilde{B_0}}$, dove $\tilde{B}_0 = [\tilde{B}_1, \ldots, \tilde{B}_n]$, che si vede essere l'ortogonalizzazione di $M_{d/1}(B)$ in 5.2.7. □

5.2.5 Estensione ad anelli ciclotomici

Si possono estendere i risultati visti anche agli anelli ciclotomici. Sia $\Omega_d = \{\xi_d^k | k \in \mathbb{Z}_d^\times\}$ l'insieme delle radici primitive dell'unità d-esime, dove $\xi_d = \exp(\frac{2\pi i}{d})$. Si denota F_d l'anello ciclotomico $\mathbb{R}[x]/\Phi_d(x)$, dove

$$\Phi_d(x) = \prod_{\xi \in \Omega_d} (x - \xi) = \prod_{k \in \mathbb{Z}_d^\times} (x - \xi_d^k) \tag{5.38}$$

è il d-esimo polinomio ciclotomico. Sia inoltre

$$\psi_d(x) = \prod_{\xi^d = 1, \xi \notin \Omega_d} (x - \xi) = \prod_{k \in \mathbb{Z}_d \setminus \mathbb{Z}_d^\times} (x - \xi_d^k) \tag{5.39}$$

Allora $x^d - 1 = \Phi_d(x) \cdot \psi_d(x)$.

Si definisce una immersione di F_d in R_d: sia e_d l'unico elemento di R_d tale che

$$e_d = 1 \mod \Phi_d \qquad e_d = 0 \mod \psi_d \tag{5.40}$$

che esiste unico per il teorema cinese del resto. Allora

$$\begin{aligned} i_d : F_d &\to R_d \\ f &\mapsto f \cdot e_d \end{aligned}$$

Quando d è chiaro dal contesto, si scrive semplicemente i. Dato $f \in F_d$ la sua immagine mediante i è l'unico elemento che soddisfa

$$i(f)(\xi) = \begin{cases} f(\xi) & \text{if } \Phi_d(\xi) = 0 \\ 0 & \text{if } \psi_d(\xi) = 0 \end{cases} \tag{5.41}$$

Tale immersione è un omomorfismo di anelli iniettivo, ed è un'isometria grazie a (5.41). La torre di anelli di R_d si ottiene per F_d semplicemente rimpiazzando i membri R_{d_i} con F_{d_i} per ogni elemento d_i nella torre di divisori di d. Sia $a \in R_d$. a appartiene all'immagine di F_d mediante i_d se e solo se $V_{d/d'}(a) \in i(F_{d'}^{d/d'})$. Questo implica che si possano definire delle applicazioni simili a M, V, ma in F_d:

$$V' = i^{-1} \circ V \circ i \qquad M' = i^{-1} \circ M \circ i$$

V' permette di passare da un elemento al successivo nella catena di anelli descritta sopra. Esse verificano che

$$V'(ab) = M'(a)V'(b) \tag{5.42}$$

Il fatto che i sia una isometria implica che anche V' lo sia. Per più dettagli si rimanda a [12].

Esempio 5.2.11. Si desidera determinare la decomposizione $LDL^\star$ e il rispettivo albero della matrice di Gram di

$$B = \begin{bmatrix} g(x) & -f(x) \\ G(x) & -F(x) \end{bmatrix}$$

dove

$$\begin{aligned} f(x) &= x^3 - x + 1 & g(x) &= 2(x^2 - x + 1) \\ F(x) &= 8(x^3 + x^2 - 1) & G(x) &= x^3 - x - 1 \end{aligned}$$

sono tutti elementi di $\mathbb{Z}[x]/(x^4 + 1)$. La matrice di Gram di B è

$$\begin{aligned} G = BB^\star &= \begin{bmatrix} g(x) & -f(x) \\ G(x) & -F(x) \end{bmatrix} \begin{bmatrix} g^\star(x) & G^\star(x) \\ -f^\star(x) & -F^\star(x) \end{bmatrix} = \\ &= \begin{bmatrix} g(x) \star g^\star(x) + f \star f^\star(x) & g(x) \star G^\star(x) + f(x) \star F^\star(x) \\ G(x) \star g^\star(x) + F(x) \star f^\star(x) & G(x) \star G^\star(x) + F(x) \star F^\star(x) \end{bmatrix} = \\ &= \begin{bmatrix} 5(2x^3 - 2x + 3) & 6x \\ -6x^3 & 65(-2x^3 + 2x + 3) \end{bmatrix} \end{aligned}$$

Innanzitutto, si applica l'algoritmo 5 per determinare le matrici L e D.

$$L = \begin{bmatrix} 1 & 0 \\ L_{2,1}(x) & 0 \end{bmatrix} \qquad D = \begin{bmatrix} D_1(x) & 0 \\ 0 & D_2(x) \end{bmatrix}$$

$$\begin{aligned} D_1(x) &= G_{1,1}(x) = 5(2x^3 - 2x + 3) \\ L_{2,1}(x) &= \frac{G_{2,1}(x)}{D_1(x)} = -\frac{6}{5}(3x^3 + 2x^2 - 2) \\ D_2(x) &= G_{2,2}(x) - L_{2,1}(x) L^\star_{2,1}(x) D_1(x) = \frac{289}{5}(-2x^3 + 2x + 3) \end{aligned}$$

Ora si attua la decomposizione $LDL^\star$ delle matrici $M_{4/2}(D_1(x))$ e $M_{4/2}(D_2(x))$.

$$G^1 = M_{4/2}(D_1(x)) = \begin{bmatrix} 15 & 10(x-1) \\ -10(x+1) & 15 \end{bmatrix} \quad G^2 = M_{4/2}(D_2(x)) = \begin{bmatrix} \frac{867}{5} & \frac{578}{5}(1-x) \\ \frac{578}{5}(1+x) & \frac{867}{5} \end{bmatrix}$$

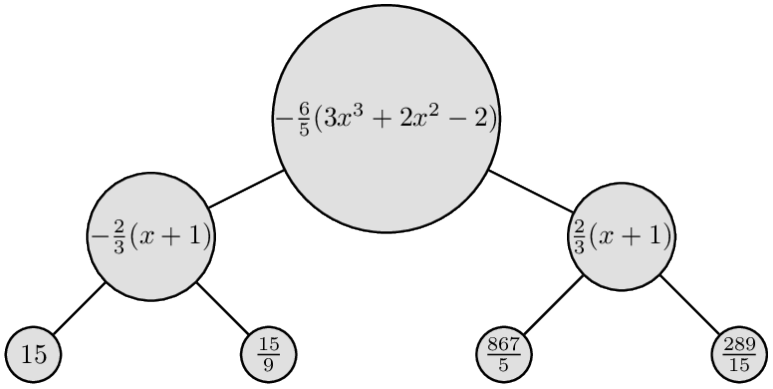
Per la prima si ottiene:

$$\begin{aligned} D^1_1(x) &= G^1_{1,1}(x) = 15 \\ L^1_{2,1}(x) &= \frac{G^1_{2,1}(x)}{D^1_1(x)} = -\frac{2}{3}(x+1) \\ D^1_2(x) &= G^1_{2,2}(x) - L^1_{2,1}(x)(L^1_{2,1})^\star(x) D^1_1(x) = \frac{15}{9} \end{aligned}$$

Per la seconda:

$$D_1^2(x) = G_{1,1}^2(x) = \frac{867}{5}$$
$$L_{2,1}^2(x) = \frac{G_{2,1}^2(x)}{D_1^2(x)} = \frac{2}{3}(1+x)$$
$$D_2^2(x) = G_{2,2}^2(x) - L_{2,1}^2(x)(L_{2,1}^2)^\star(x)D_1^2(x) = \frac{289}{15}$$

L'albero risultante è



Capitolo 6

FALCON

In questo capitolo si descrive FALCON, uno schema di firma digitale che prende forma a partire dal GPV Framework sui reticoli NTRU, usando un campionamento nuovo da reticoli, detto Fast Fourier Sampling.

È uno dei tre schemi di firma vincitori della sfida indetta dal NIST per sostituire gli schemi attuali.

Sono stati scelti come base i reticoli, perché riescono a garantire dimensione di chiavi e firme bassa, soprattutto per reticoli strutturati, come quelli NTRU.

La scelta del GPV framework come base teorica è motivata dal fatto che è stato dimostrato in [7] che questa struttura è resistente sia agli attacchi dei computer classici, che di quelli quantistici. Specializzandosi a reticoli NTRU, si riesce facilmente ad adattare le dimostrazioni, mantenendo sicurezza.

Infine, il nuovo campionatore di trapdoor Fast Fourier è stato scelto perché è stato dimostrato essere asintoticamente veloce come il più veloce di questi algoritmi e inoltre sicuro quanto il più sicuro.

È interessante notare che si possono fare altre scelte per Trapdoor Sampler e struttura dei reticoli, mantenendo intatto lo scheletro del GPV framework, ad esempio si può utilizzare il campionamento Nearest Plane di Klein, ottenendo maggior velocità, ma minor sicurezza.

I primi a coniugare NTRUSign con il framework GPV furono Stehlé e Steinfeld in [25]. Inoltre, essi si interessarono a dimostrare rigorosamente la sicurezza del crittosistema NTRUEncrypt e dello schema di firma NTRUSign da loro modificati, basandola su problemi provati difficili sui reticoli. Nonostante si possano basare su altri problemi sui reticoli dei crittosistemi sicuri, essi hanno una complessità tempo e spazio quadratica nel parametro di sicurezza N, il che li rende impraticabili. Per questo, l'obiettivo è rendere sicuro NTRU, che, grazie alla struttura dei reticoli, dà firme e chiavi pubbliche che sono più corte ed efficienti. Ducas, Lyubashevsky e Prest si occuparono in [10] di rendere lo schema implementabile dal punto di vista pratico, mentre i primi si occuparono più della parte teorica.

Le modifiche principali allo schema sono le seguenti:

- Si sostituisce a $\frac{\mathbb{Z}[x]}{x^N-1}$ l'anello $\frac{\mathbb{Z}[x]}{x^N+1}$, dove N è sempre una potenza di 2. Con tale scelta di N, il polinomio x^N+1 è irriducibile in $\mathbb{Q}$, ha radici distinte in $\mathbb{C}$ ed è un polinomio ciclotomico: infatti si scrive come

$$x^N + 1 = \prod_{j \in (\mathbb{Z}^{2N})^\times} (x - \omega^j) \tag{6.1}$$

  Questa scelta è giustificata dalla maggior sicurezza e resistenza alle collisioni dimostrata da alcuni studiosi riguardo le funzioni di hash su $\frac{\mathbb{Z}[x]}{x^N+1}$.

- Si sceglie $q = 1 \mod 2N$ primo, affinché $x^N + 1$ spezzi in fattori lineari modulo q, infatti con tale scelta, visto che $\gcd(q, 2N) = 1$, x^N+1 si scompone in $\frac{\phi(2N)}{d} = \frac{N}{d}$ fattori monici irriducibili, dove d è il minimo intero positivo tale che $q^d = 1 \mod 2N$, cioè $d = 1$. Per una dimostrazione di questo fatto si rimanda a [21].

- Si modifica la generazione delle chiavi private: $f(x)$ e $g(x)$ vengono entrambi campionati da una distribuzione Gaussiana discreta, centrata nell'origine e avente approssimativamente standard deviation uguale a $1.17(q/2)^{1/2}$, rigettati se non invertibili. Questo perché viene dimostrato da Stehlé e Steinfeld un risultato secondo il quale in questo modo la chiave pubblica $h(x)$ sarà distribuita uniformemente su $\mathbb{Z}_q[x]/(x^N+1)$, affinché l'attaccante non possa ottenere informazioni aggiuntive.

Sia $R = \frac{\mathbb{Z}[x]}{x^N+1}$, in cui N è una potenza di 2. Dato $f(x)$ un elemento di R, la matrice con righe i vettori dei coefficienti di $f(x)$ e le sue rotazioni è

$$C(f(x)) = \begin{bmatrix} c(f(x)) \\ c(x \star f(x)) \\ \vdots \\ c(x^{N-1} \star f(x)) \end{bmatrix} = \begin{bmatrix} f_0 & f_1 & \dots & f_{N-1} \\ -f_{N-1} & f_0 & \dots & f_{N-2} \\ \ddots & \ddots & \ddots & \ddots \\ -f_1 & -f_2 & \dots & f_0 \end{bmatrix}$$

Si tratta di una matrice anticircolante. Siano $f(x), g(x) \in R$. Allora è facile mostrare che

$$C(f(x)) + C(g(x)) = C(f(x) + g(x))$$
$$C(f(x)) \cdot C(g(x)) = C(f(x) \star g(x))$$

Sia $h(x) = g(x) \star f^{-1}(x) \mod q$. Il **reticolo NTRU** associato a $h(x)$ e q su R è

$$\Lambda_{h,q} = \{(u(x), v(x)) \in R^2 : u(x) + v(x) \star h(x) = 0 \mod q\}$$

È il reticolo generato dalle righe della matrice $B_{h,q} = \begin{bmatrix} -C(h(x)) & I_N \\ qI_N & 0_N \end{bmatrix}$ di dimensione $2N$. Infatti, un elemento del reticolo generato da tale matrice è dato

da

$$(a(x), b(x)) \cdot \begin{bmatrix} -C(h(x)) & I_N \\ qI_N & 0_N \end{bmatrix} = (-a(x) \star h(x) + qb(x), a(x)) = (u(x), v(x))$$

e perciò

$$u(x) + v(x) \star h(x) = -a(x) \star h(x) + qb(x) + a(x) \star h(x) = qb(x) = 0 \mod q$$

Per memorizzare $B_{h,q}$ è sufficiente tenere in memoria $h(x)$.

Definizione 6.0.1. *Sia $f(x) \in \mathbb{Q}[x]/(x^N + 1)$. Si denota con $f^\star(x)$ l'**aggiunto** di $f(x)$, l'unico elemento dell'anello tale che*

$$f^\star(x) = f\left(\frac{1}{x}\right) = f_0 - \sum_{i=1}^{N-1} f_{N-i} x^i$$

Esso è tale che $C(f^\star(x)) = C(f(x))^T$.

6.1 GPV Framework

Si riassume brevemente in cosa consiste il GPV framework, trattato nel capitolo 4. Siano n, m, q interi positivi, con $m > n$.

La **chiave pubblica** è una matrice $A \in \mathbb{Z}_q^{m \times n}$, la cui trasposta generi (4.48) e che definisca un ulteriore reticolo (4.47), come nucleo dell'applicazione lineare associata modulo q.

La **chiave privata** è la base del reticolo (4.47), che è il duale del reticolo generato da A^T, moltiplicato per q, perciò $B \cdot A = 0 \mod q$.

Sia $H : \{0,1\}^* \to \mathbb{Z}_q^n$ una funzione di hash, dove $\{0,1\}^*$ indica $\bigcup_{n \in \mathbb{N}} \{0,1\}^n$. Dato un messaggio $\underline{m} \in \{0,1\}^*$, una sua **firma** $\underline{s}$ è un elemento di norma piccola in $\mathbb{Z}_q^m$ tale che

$$\underline{s}A = H(\underline{m})$$

Per computare $\underline{s}$, dapprima si calcola una soluzione generica dell'equazione mediante algebra lineare, poi vi si sottrae un elemento vicino del reticolo (4.47) per ottenere una soluzione corta; infatti, il prodotto di tale elemento del reticolo con A è nullo. La maniera in cui si calcola tale elemento del reticolo distingue il framework GPV da NTRUSign: nel secondo caso si utilizza l'algoritmo di Babai illustrato al capitolo 1, che alla lunga rivela informazioni circa la base segreta, mentre il primo evita ciò utilizzando una variante randomizzata dell'algoritmo di Klein.

Dato che H è pubblica, la **verifica** consiste semplicemente nel sincerarsi che la firma sia soluzione dell'equazione sopra.

Si consideri $N = 2^k$, q un intero e $R = \frac{\mathbb{Z}[x]}{x^N+1}$. Un reticolo NTRU su R è determinato da polinomi $f(x), g(x), F(x), G(x) \in R, h(x) \in R_q$ che verifichino

$$f(x) \star G(x) - g(x) \star F(x) = q \in R \qquad h(x) = g(x) \star f^{-1}(x) \mod q \tag{6.2}$$

La **chiave pubblica** in questo caso è

$$A = \begin{bmatrix} 1 \\ h(x) \end{bmatrix} \in R_q^2. \tag{6.3}$$

Conoscere A è equivalente a conoscere il polinomio $h(x) \in R_q$.

La **chiave privata** è

$$B = \begin{bmatrix} g(x) & -f(x) \\ G(x) & -F(x) \end{bmatrix} \tag{6.4}$$

Questo perché la rispettiva matrice delle rotazioni di tali vettori è ancora base del reticolo $\Lambda_{h,q}$, dove $h(x)$ è il rapporto modulo q tra $g(x)$ e $f(x)$ (ciò si prova vedendo che la matrice di cambio di base ha determinante di modulo 1 e entrate in $\mathbb{Z}$). Le matrici A e B hanno prodotto nullo modulo q, infatti

$$\begin{aligned} BA &= \begin{bmatrix} g(x) & -f(x) \\ G(x) & -F(x) \end{bmatrix} \cdot \begin{bmatrix} 1 \\ h(x) \end{bmatrix} = \\ &= (g(x) - f(x) \star h(x), G(x) - F(x) \star h(x)) \overset{(6.2)}{=} 0 \mod q \end{aligned}$$

Randomizzazione hash

Un problema che occorre in questo contesto è che se vengono pubblicate due firme a partire dallo stesso messaggio, questo inficia la sicurezza. La soluzione proposta da FALCON consiste nel concatenare alla stringa del messaggio $\underline{m}$ un elemento $\underline{r} \in \{0,1\}^k$ casuale prima di applicarvi la funzione di hash H. Se $\underline{r}$ è sufficientemente lungo, si eviteranno collisioni anche firmando due volte il medesimo messaggio. Questa soluzione è semplice e rimane sicura per il GPV framework.

Il livello di sicurezza dello schema è λ se sono necessarie in media 2^λ operazioni per rompere lo schema. Un altro parametro è il numero massimo q_s di firme richiedibili, senza dare informazioni aggiuntive all'attaccante, che facilitino la rottura.

Si sceglie $k = \lambda + \log_2(q_s)$, affinché la probabilità di collisione sia inferiore a $q_s 2^{-\lambda}$. Il livello di sicurezza maggiore richiesto dal NIST è 256, e il numero massimo di firme producibile si attesta per il NIST a 2^{64}, perciò $k = 320$.

La firma consiste della coppia $(s_0(x), s_1(x))$, di norma piccola e che è soluzione in $R_q = \mathbb{Z}_q[x]/(x^N + 1)$ di

$$s_0(x) + s_1(x) \star h(x) = H(\underline{r}||\underline{m})$$

dove con $||$ si intende l'operatore di concatenazione. Chi firma pubblica solamente $(\underline{r}, s_1(x))$, dato che H è pubblica, quindi $s_0(x)$ si ricava dall'equazione sopra.

Per verificare la firma si calcola dapprima $s_0(x)$ e si accerta che la norma del vettore $(s_0(x), s_1(x))$ sia inferiore ad un limite prestabilito β, altrimenti la firma è giudicata falsa.

6.2 Parametri

Gli autori suggeriscono i parametri da scegliere per istanziare lo schema, essi sono tutti pubblici.

N Si sceglie una potenza di 2, solitamente $N = 512$ oppure $N = 1024$.

q Si sceglie il più piccolo q primo e della forma $2kN + 1$, cioè

$$q = 12 \cdot 1024 + 1 = 12289. \tag{6.5}$$

Tale valore è abbastanza grande da evitare attacchi al relativo SIS.

$\|\mathbf{B}\|_{\mathbf{GS}}$ In [10] si enuncia un'euristica che garantisce che $\|B\|_{GS}$ venga minimizzata quando $\|(g(x), -f(x))\| \approx 1.17\sqrt{q}$, perciò si generano i coefficienti di $f(x)$ e $g(x)$ da una distribuzione Gaussiana discreta centrata in 0 e di standard deviation circa $1.17\sqrt{q/2N}$, si calcola $\|B\|_{GS}$ e si ricampiona se è maggiore di $1.17\sqrt{q}$.

Standard deviation delle firme σ Le firme vengono campionate attraverso il Fast Fourier sampling da una distribuzione Gaussiana discreta con deviazione standard

$$\sigma = \frac{1}{\pi}\sqrt{\frac{\log(4n(1+\frac{1}{\epsilon}))}{2}} \cdot 1.17\sqrt{q} \geq \eta_\epsilon(\mathbb{Z}^{2N}) \cdot \|B\|_{GS} \tag{6.6}$$

dove

$$\epsilon \leq \frac{1}{q_s \lambda} = \frac{1}{2^{64} \cdot 256}. \tag{6.7}$$

Bound sulla norma delle firme Le firme devono verificare la seguente disuguaglianza:

$$\|(s_0(x), s_1(x))\|^2 \leq \lfloor \beta^2 \rfloor \tag{6.8}$$

Una scelta per β può essere $\beta = 1.1 \cdot \sigma\sqrt{2N}$; il valore atteso della norma della firma $\underline{s} = (s_0(x), s_1(x))$ è $\sigma\sqrt{2N}$, quindi ogni firma con di norma maggiore viene rifiutata. In questo modo, la probabilità che una firma prodotta in maniera corretta venga scartata è molto bassa.

6.3 Preliminari teorici

6.3.1 FFT e NTT

Il polinomio $x^N + 1$ ha radici $\left\{\omega_k = \exp\left(\frac{i(2k+1)\pi}{N}\right) \middle| 0 \leq k < N\right\}$, radici primitive dell'unità $2N$-esime. Si può estendere la definizione di **FFT** anche a R valutando i suoi elementi in tali valori; l'ordine è indifferente, ma è importante rimanere coerenti mantenendo sempre il medesimo. Dunque, la FFT di $f(x)$ è

$$\mathrm{FFT}(f(x)) = (f(\omega_k))_{k=0\ldots,N-1} \tag{6.9}$$

La **NTT** (Number Theoretic Transform) è l'analogo della FFT su $\mathbb{Z}_p$, con p un primo uguale a 1 modulo $2N$. Tale congruenza garantisce che $x^N + 1$ abbia N radici distinte in $\mathbb{Z}_p$, e perciò la NTT di un elemento $f(x)$ di R è il vettore delle valutazioni di $f(x)$ in tali radici.

Osservazione 6.3.1. Si ha la seguente torre di anelli:

$$\mathbb{Q} \subseteq \mathbb{Q}[x]/(x^2+1) \subseteq \cdots \subseteq \mathbb{Q}[x]/(x^{N/2}+1) \subseteq \mathbb{Q}[x]/(x^N+1) \tag{6.10}$$

Inoltre, dividendo i polinomi nei loro coefficienti di indice pari e dispari si ottiene la seguente catena di isomorfismi:

$$\mathbb{Q}^N \cong (\mathbb{Q}[x]/(x^2+1))^{N/2} \cong \cdots \cong (\mathbb{Q}[x]/(x^{N/2}+1))^2 \cong \mathbb{Q}[x]/(x^N+1) \tag{6.11}$$

La torre e la catena rimangono valide sostituendo $\mathbb{Z}$ a $\mathbb{Q}$. Dotando tali moduli del prodotto interno definito sotto, la catena è di isomorfismi di anelli.

$$\langle a(x), b(x)\rangle = \frac{1}{N}\sum_{\xi^N=-1} a(\xi)\overline{b(\xi)} \qquad a(x), b(x) \in R \tag{6.12}$$

Per la formula di Parseval, tale prodotto interno corrisponde al prodotto scalare usuale sul vettore dei coefficienti. Esso induce una norma, che corrisponde alla norma euclidea. A seconda della rappresentazione che si ha del polinomio, sarà più conveniente usare l'una o l'altra forma.

6.3.2 Splitting e Merging

Si tratta degli isomorfismi che legano due anelli consecutivi della catena di isomorfismi descritti sopra. Si tratta di $V_{d/d'}$ e la sua inversa, ristretti al caso in cui d sia una potenza di 2. Si tratta sostanzialmente del procedimento che si attua durante la FFT. Nel dettaglio, sia $f(x) = \sum_{i=0}^{N-1} a_i x^i$, dove $N = 2^k$. Allora esso si

può suddividere tra monomi con potenze pari e con potenze dispari:

$$\begin{aligned} f(x) &= \sum_{i=0}^{N/2-1} a_{2i}x^{2i} + \sum_{i=0}^{N/2-1} a_{2i+1}x^{2i+1} = \\ &= \sum_{i=0}^{N/2-1} a_{2i}x^{2i} + x\sum_{i=0}^{N/2-1} a_{2i+1}x^{2i} = f_0(x^2) + xf_1(x^2) \end{aligned}$$

Quindi

$$\text{split}(f(x)) = (f_0(y), f_1(y)) \tag{6.13}$$

dove $y = x^2$. Esso percorre la catena da destra verso sinistra. Sfruttando che

$$f(-x) = f_0(x^2) - xf_1(x^2) \tag{6.14}$$

si possono ricavare $f_0(y)$ e $f_1(y)$ da:

$$f(x) + f(-x) = 2f_0(x^2) \qquad f(x) + f(-x) = 2xf_1(x^2) \tag{6.15}$$

L'inversa di split è merge. Dati $f_0(x), f_1(x) \in \mathbb{Q}[x]/(x^{N/2}+1)$, essi possono formare un polinomio in $\mathbb{Q}[x]/(x^N+1)$ contribuendo rispettivamente ai coefficienti di potenze pari e ai coefficienti di potenze dispari.

$$\text{merge}(f_0(x), f_1(x)) = f_0(x^2) + xf_1(x^2) \tag{6.16}$$

Questi due isomorfismi sono utilizzati all'interno dei diversi algoritmi dello schema, ma si operano sulla FFT degli elementi.

L'obiettivo è rendere compatibile la rappresentazione con le operazioni dell'anello.

Ad esempio siano $a(x), b(x), c(x) \in \mathbb{Q}[x]/(x^N+1)$ tali che

$$a(x) = b(x) \star c(x) = C_c(b(x)) \tag{6.17}$$

dove C_c è l'omomorfismo di moltiplicazione per $c(x)$. Su $\mathbb{Q}[x]/(x^{N/2}+1)$ si ottiene

$$\begin{bmatrix} a_0(y) & a_1(y) \end{bmatrix} = \begin{bmatrix} b_0(y) & b_1(y) \end{bmatrix} \begin{bmatrix} c_0(y) & c_1(y) \\ y \star c_1(y) & c_0(y) \end{bmatrix} \tag{6.18}$$

Infatti

$$\begin{aligned} b(x) \star c(x) &= (b_0(x^2) + xb_1(x^2)) \star (c_0(x^2) + xc_1(x^2)) = \\ &= b_0(x^2) \star c_0(x^2) + x^2 \star b_1(x^2) \star c_1(x^2) + x \star (b_0(x^2) \star c_1(x^2) + b_1(x^2) \star c_0(x^2)) = \\ &= b_0(y) \star c_0(y) + b_1(y) \star c_1(y) \star y + x \star (b_0(y) \star c_1(y) + b_1(y) \star c_0(y)) \end{aligned} \tag{6.19}$$

Perciò

$$M_{N/\frac{N}{2}}(c(x)) = \begin{bmatrix} c_0(y) & c_1(y) \\ y \star c_1(y) & c_0(y) \end{bmatrix} \tag{6.20}$$

rappresenta la moltiplicazione per $c(x)$ in $\mathbb{Q}[x]/(x^{N/2}+1)$.

Per passare da un anello all'altro della catena si utilizza la norma del campo, definita da un campo $\mathbb{L}$ ad un suo sottocampo $\mathbb{K}$; essa è definita come segue: sia $k \geq 1$, $\mathbb{L} = \mathbb{Q}[x]/(x^{2^k}+1)$ e $\mathbb{K} = \mathbb{Q}[y]/(y^{2^{k-1}}+1)$

$$\begin{aligned} N_{\mathbb{L}/\mathbb{K}} : \mathbb{L} &\to \mathbb{K} \\ f(x) &\mapsto f_0^2(y) - y f_1^2(y) \end{aligned} \tag{6.21}$$

dove $\text{split}(f(x)) = (f_0(y), f_1(y))$. Osservando che $y = x^2$, si può definire equivalentemente

$$N_{\mathbb{L}/\mathbb{K}}(f) = f(x)f(-x)\big|_{x^2=y} \tag{6.22}$$

6.4 Chiavi

La prima parte della generazione delle chiavi consiste nel determinare delle soluzioni $F(x), G(x)$ di norma piccola della cosiddetta equazione NTRU, la seconda nel determinare un albero FALCON, ossia un albero binario correlato con la decomposizione GSO della base trovata.

6.4.1 Generazione delle chiavi

L'algoritmo di generazione delle chiavi prende in input i parametri N, q come nella versione classica, a cui si aggiunge un parametro $s > 0$: la novità di questa procedura, proposta da Steinfeld e Stehlé, è il campionamento di $f(x), g(x)$ da una distribuzione Gaussiana discreta centrata in 0 e con deviazione standard s mediante l'algoritmo descritto da Gentry, Peikert e Vaikuntanathan.

Si determinano poi $F(x), G(x)$ tali che

$$f(x) \star G(x) - g(x) \star f(x) = q$$

affinché si possa calcolare una base segreta

$$B_{f,g} = \begin{bmatrix} C(g(x)) & -C(f(x)) \\ C(G(x)) & -C(F(x)) \end{bmatrix}$$

Campionando le chiavi in tal modo, la norma dei vettori relativi è proporzionale a $\|B_{f,g}\|_{GS}$, cioè la massima norma dei vettori della GSO di $B_{f,g}$. Visto che si desidera mantenere questa norma piccola affinché il campionamento dal reticolo $\Lambda_{h,q}$ sia buono, l'obiettivo è selezionare $f(x), g(x)$ che minimizzino tale quantità. La cosa interessante è che questo valore può essere calcolato senza operare l'ortogonalizzazione, riducendo notevolmente la complessità computazionale del processo.

Lemma 6.4.1. *Siano $\underline{b}_1, \ldots, \underline{b}_{2N}$ le righe di $B_{f,g}$. Allora*

$$\|B_{f,g}\|_{GS} = \max\{\|\tilde{\underline{b}}_1\|_2, \|\tilde{\underline{b}}_{N+1}\|_2\} \tag{6.23}$$

Dimostrazione. Siano π_V l'operatore di proiezione ortogonale su V e r l'isometria su $\mathbb{R}^{2N}$ che, a partire da $(f(x), g(x))$, restituisce $(x \star f(x), x \star g(x))$, $V_i = \text{span}(\underline{b}_1, \ldots, \underline{b}_i)^\perp$. Si può scrivere che per ogni $1 \leq i \leq N$

$$\underline{b}_i = r^{i-1}(\underline{b}_1) \qquad \underline{b}_{N+i} = r^{i-1}(\underline{b}_{N+1}),$$

inoltre per ogni $i = 1, \ldots, 2N$

$$\tilde{\underline{b}}_i = \pi_{V_{i-1}}(\underline{b}_i)$$

In generale, sono vere le seguenti:

$$\|\pi_V(\underline{b})\|_2 \leq \|\underline{b}\|_2$$

$$V \subseteq W \Longrightarrow \|\pi_V(\underline{b})\|_2 \leq \|\pi_W(\underline{b})\|_2$$

Applicandole al caso in esame si ottiene che quando $1 \leq i \leq N$

$$\|\tilde{\underline{b}}_i\|_2 \leq \|\underline{b}_i\|_2 = \|r^{i-1}(\underline{b}_1)\|_2 = \|\underline{b}_1\|_2 = \|\tilde{\underline{b}}_1\|_2 \tag{6.24}$$

Poiché $V_N^\perp$ è stabile rispetto all'azione di r, lo deve essere anche V_N, perciò

$$\pi_{V_N}(\underline{b}_{N+i}) = \pi_{V_N}(r^{i-1}(\underline{b}_{N+1})) = r^{i-1}(\pi_{V_N}(\underline{b}_{N+1})) = r^{i-1}(\tilde{\underline{b}}_{N+1})$$

da cui si ottiene che, visto che $V_{N+i-1} \subseteq V_N$,

$$\|\tilde{\underline{b}}_{N+1}\|_2 = \|r^{i-1}(\tilde{\underline{b}}_{N+1})\|_2 = \|\pi_{V_N}(\underline{b}_{N+i})\|_2 \geq \|\pi_{V_{N+i-1}}(\underline{b}_{N+i})\|_2 = \|\tilde{\underline{b}}_{N+i}\|_2. \tag{6.25}$$

□

Prima dell'ortogonalizzazione le prime N righe della basis matrix hanno norma $\|(g(x), -f(x))\|$ e le rimanenti $\|(G(x), -F(x))\|$. Dopo, tali norme diminuiscono e assumono più di due valori, ma i massimi vengono raggiunti dall'ortogonalizzazione dei vettori originali, cioè non ruotati. Il primo vettore GSO è $\tilde{\underline{b}}_1 = \underline{b}_1$, mentre l'N+1-esimo si può scrivere senza operare il processo di ortogonalizzazione, in funzione di $q, f(x), g(x)$ e i loro coniugati.

Lemma 6.4.2.

$$\|\tilde{\underline{b}}_{N+1}\|_2 = \left\| \left(\frac{q f^\star(x)}{f \star (x) f^\star(x) + g(x) \star g^\star(x)}, \frac{q g^\star(x)}{f(x) \star f^\star(x) + g(x) \star g^\star(x)} \right) \right\|$$

Dimostrazione. [10, Lemma 3]. □

In questo modo, è possibile calcolare $\|B_{f,g}\|_{GS}$ semplicemente a partire da $f(x), g(x)$ e decidere se è necessario ricampionarli. Un controllo preliminare prima di calcolare $\|\tilde{b}_{n+1}\|_2$ è il seguente:

Lemma 6.4.3.

$$\|\tilde{\underline{b}}_{N+1}\|_2 \geq \frac{q}{\|\underline{b}_1\|_2} \tag{6.26}$$

Dimostrazione. Per (1.20) e (2.12) si ha che

$$q^N = |\det(B_{f,g})| = |\det(\tilde{B}_{f,g})| = \prod_{i=1}^{2N} \|\tilde{\underline{b}}_i\|$$

Sfruttando (6.24) e (6.25), si ottiene che

$$q^N \leq \|\tilde{\underline{b}}_1\|_2^N \|\tilde{\underline{b}}_{N+1}\|_2^N$$

da cui la tesi. □

Si verifica sperimentalmente che $\|\tilde{b}_{N+1}\|_2$ si avvicina molto alla sua minorazione, per cui, per minimizzare $\|B_{f,g}\|_{GS}$ si devono minimizzare contemporaneamente $\|\underline{b}_1\|_2$ e $q/\|\underline{b}_1\|_2$. La scelta ottimale di $\|\underline{b}_1\|_2$ è circa $\sqrt{\frac{qe}{2}} \approx 1.1658\sqrt{q}$, e deriva da un'euristica presentata in [10], confermata dagli esperimenti. Affinché l'algoritmo di generazione delle firme termini più velocemente, si sceglie $\|\underline{b}_1\|_2 \approx 1.17\sqrt{q}$.

La versione concreta del teorema 4.2.4 data da [11] è

Teorema 6.4.4. *Siano n e λ due interi positivi, e sia $\epsilon = 2^{-\lambda}/(2n)$. Per ogni reticolo di dimensione n con basis matrix B, per ogni target $\underline{c} \in \mathbb{Z}^n$ e $\sigma \geq \|B\|_{GS}\eta_\epsilon(\mathbb{Z})$, (dove $\eta_\epsilon(\mathbb{Z}) \approx \frac{1}{\pi}\sqrt{\frac{1}{2}\log\left(2+\frac{2}{\epsilon}\right)}$) l'algoritmo 4 fornisce campioni da una distribuzione con distanza statistica inferiore a $2^{-\lambda}$ da $D_{\mathbb{Z},\sigma,c}$.*

Nel caso in esame $n = 2N$, perciò $\epsilon = \frac{2^{-\lambda}}{4N}$.

$c(f(x))$ e $c(g(x))$ vengono campionati da una distribuzione Gaussiana discreta su $\mathbb{Z}^N$ centrata in 0 e con deviazione standard $1.17\sqrt{q/2}$, affinché il valore atteso della norma del vettore $(c(f(x)), c(g(x)))$ sia $1.17\sqrt{q}$.

Sfruttando che una distribuzione normale N-dimensionale con matrice delle covarianze diagonale si può vedere come N normali indipendenti unidimensionali, una maniera di ottenere $c(f(x))$ e $c(g(x))$ è campionare ogni entrata f_i e g_i dalla distribuzione $D_{\mathbb{Z},1.17\sqrt{q/2N}}$.

Una maniera più semplice per verificare che $f(x)$ sia invertibile modulo q è mediante la NTT. $f(x)$ è invertibile in R_q se $\gcd(f(x), x^N + 1) = 1$ in $\mathbb{Z}_q$, cioè se i due polinomi non hanno fattori comuni, e quindi radici in comune. Perciò $f(x)$ è invertibile in R_q se e solo se nessuna delle entrate di NTT(f(x)) è nulla. Si ricampiona $f(x)$ finché non è invertibile modulo q.

Dopodiché, si calcola la corrispondente $\|B_{f,g}\|_{GS}$ computando

$$\max\left\{\|(g(x), -f(x))\|, \left\|\left(\frac{qf^\star(x)}{f(x)\star f^\star(x) + g(x)\star g^\star(x)}, \frac{qg^\star}{ff^\star + gg^\star}\right)\right\|\right\}$$

Se tale valore è meggiore di $1.17\sqrt{q}$ si deve ricominciare, altrimenti il processo termina.

Equazione NTRU

Per determinare una soluzione in R di

$$f(x) \star G(x) - g(x) \star F(x) = q \tag{6.27}$$

si illustra una nuova tecnica che sfrutta la struttura di R.

Si tratta di un algoritmo ricorsivo che si riduce a calcolare la soluzione in $\mathbb{Z}$.

Algorithm 9 Algoritmo di risoluzione dell'equazione NTRU $NTRUSolve_{N,q}$

Input: $f(x), g(x) \in R$
Output: $F(x), G(x)$ soluzioni di (6.27)

```
if N = 1 then
Determinare u, v ∈ Z tali che uf − vg = gcd(f, g)
    if gcd(f, g) ≠ 1 then return ∅
    end if
    (F, G) ← (vq, uq)
     return (F, G)
else
    f'(x) ← N(f(x))
    g'(x) ← N(g(x))
    (F'(x), G'(x)) ← NTRUSolve_{N/2,q}(f'(x), g'(x))
    F(x) ← F'(x^2)g(−x)
    G(x) ← G'(x^2)f(−x)
     Reduce(f(x), g(x), F(x), G(x))
end if
    return (F(x), G(x))
```

L'algoritmo funziona perché dopo la chiamata ricorsiva, si moltiplicano $F'(x)$ e $G'(x)$ per i fattori che rendono vera l'equazione con $f(x), g(x)$: si supponga che sia vera la seguente

$$f'(y)G'(y) - g'(y)F'(y) = q \text{ in } \mathbb{Z}[y]/(y^{N/2} + 1)$$

Sia $y = x^2$, ricordando la definizione di $f'(y)$ e $g'(y)$ e vedendo la stessa equazione in $\mathbb{Z}[x]/(x^N + 1)$ si ha

$$f(x)f(-x)G'(x^2) - g(x)g(-x)F'(x^2) = q$$

da cui si ottiene che

$$G(x) = f(-x)G'(x^2) \qquad F(x) = g(-x)F'(x^2)$$

Una volta ottenuti $F(x), G(x)$, non è detto che abbiano norma piccola, quindi si applica una procedura che li riduce mediante la sottrazione del multiplo più

vicino di $(f(x), g(x))$, come in NTRUSign. Esso è

$$k(x) = \left\lfloor \frac{f^\star(x) \star F(x) + g^\star(x) \star G(x)}{f(x) \star f^\star(x) + g(x) \star g^\star(x)} \right\rceil$$

Questa procedura si può ripetere fino a che $k(x) = 0$.

Una volta calcolati i polinomi $f(x), g(x), F(x), G(x)$, visto che il processo di calcolo è piuttosto costoso, vanno conservati in memoria almeno tre di questi, cosicché sia semplice calcolare il quarto mediante l'equazione NTRU. Infine, la chiave pubblica $h(x)$ è data dal rapporto modulo q tra $g(x)$ e $f(x)$.

6.4.2 Albero FALCON

Si tratta di un albero binario definito induttivamente:

- Un albero FALCON di altezza 0 è un numero reale positivo.
- Un albero di altezza $k > 0$ è tale che la sua radice sia un polinomio in $\mathbb{Q}[x]/(x^N + 1)$, dove $N = 2^k$ e i suoi figli alberi FALCON di altezza $k - 1$.

Per diminuire la complessità computazionale del processo, il valore dei nodi interni è rappresentato mediante FFT, ottenendo numeri complessi. La procedura si può attuare anche mantenendo la rappresentazione in polinomi, solo con un costo computazionale maggiore. L'albero viene rappresentato da $2^k(1 + k)$ elementi. Il contenuto di questo albero è determinato a partire da $f(x), g(x), F(x), G(x)$. Data la base segreta

$$B = \begin{bmatrix} g(x) & -f(x) \\ G(x) & -F(x) \end{bmatrix} \in R^{2\times 2},$$

si calcola la decomposizione LDL* di $BB^\star$. Per utilizzare il Fast Fourier Sampling, non è sufficiente possedere L.

$$L = \begin{bmatrix} 1 & 0 \\ L_{1,0}(x) & 1 \end{bmatrix}$$

L è una matrice triangolare inferiore unitaria avente 4 entrate, perciò l'unico valore non noto a priori è l'entrata in basso a sinistra, che quindi è l'unica che si conserva in memoria.

$L_{1,0}(x)$ è la radice dell'albero, si utilizzano in seguito l'operatore M definito sopra per ottenere a partire da ogni elemento diagonale di D delle matrici per costruire dei sottoalberi ricorsivamente mediante la decomposizione LDL*. La ricorsione termina quando la matrice considerata ha entrate in $\mathbb{Q}$. È importante ricordare che ad ogni livello dell'albero corrispondono elementi in un anello diverso. Le foglie sono gli elementi non nulli delle ultime matrici diagonali ottenute, cioè con entrate in $\mathbb{Q}$.

Osservazione 6.4.1. La rappresentazione mediante FFT di un polinomio reale $f(x)$ ha metà delle componenti ridondanti, perché se ξ è radice la è anche la sua coniugata $\bar{\xi}$, e inoltre $f(\bar{\xi}) = \overline{f(\xi)}$. Perciò sono sufficienti $N/2$ numeri complessi per rappresentare $\text{FFT}(f(x))$.

Una volta applicato l'algoritmo visto nel capitolo 6 per la decomposizione Fast Fourier LDL*, si ottiene un albero. Per arrivare all'albero FALCON resta solo da normalizzare i valori delle foglie: il valore è sostituito dal rapporto tra σ e la radice quadrata di tale quantità; questo ha a che fare col campionamento da effettuare. In effetti, visto che la decomposizione LDL* viene ottenuta a partire dalla GSO, le entrate delle ultime matrici diagonali sono le norme al quadrato dei vettori ortogonalizzati. Come nell'algoritmo di Klein, il campionamento di interi si effettua mediante una distribuzione Gaussiana discreta su $\mathbb{Z}$ con standard deviation quella iniziale divisa per la norma del vettore ortogonalizzato corrispondente, cioè la radice quadrata dei valori delle foglie.

6.5 Firma e verifica

Per firmare un documento $\underline{m} \in \{0,1\}^*$, per prima cosa si campiona $\underline{r} \in \{0,1\}^{320}$ uniformemente e lo si concatena a $\underline{m}$. Poi si applica una funzione di hash H a $(\underline{r}||\underline{m})$ che lo renda un elemento $c(x)$ di $\mathbb{Q}[x]/(x^N+1)$, o equivalentemente il vettore dei suoi coefficienti $c(c(x))$. Si calcola una prima controimmagine $(t_0(x), t_1(x))$ di $c(x)$ rispetto alla funzione che moltiplica a destra per A^T, e poi si sfrutta la conoscenza della base segreta per determinare una coppia $(s_0(x), s_1(x))$ di norma piccola tale che

$$s_0(x) + s_1(x) \star h(x) = c(x) \mod q \tag{6.28}$$

Il vettore dei coefficienti rispetto alla base segreta di una possibile controimmagine di $c(x)$ è $(t_0(x), t_1(x)) = (c(x), 0)B_{f,g}^{-1}$: infatti

$$c(x) = (c(x),0)_1 + (c(x),0)_2 \star h(x) = (c(x),0) \cdot A = (t_0(x), t_1(x)) \cdot B_{f,g} \cdot A \tag{6.29}$$

Per non far trapelare informazioni sulla base segreta, si deve utilizzare un generatore di trapdoor. A differenza del GPV framework, se ne utilizza uno inedito, detto Fast Fourier Sampling. Si tratta di una variante randomizzata dell'algoritmo Fast Fourier Nearest Plane visto nel capitolo precedente, cioè una versione ricorsiva dell'algoritmo di Klein. In input all'algoritmo si dà l'albero FALCON costruito nel processo di generazione delle chiavi, e una controimmagine $(t_0(x), t_1(x))$ dell'hash del messaggio. L'algoritmo attua ricorsivamente la procedura usando i nodi dell'albero prima su $\underline{t}_1(x)$ e poi su $\underline{t}_0(x)$, tenendo conto dell'output finale di $\underline{t}_1(x)$: per questo le due ricorsioni non possono essere effettuate in parallelo. Tale scelta è motivata dalla volontà di nascondere la struttura della base privata agli attaccanti.

Si ha

$$\begin{aligned}(t_0(x), t_1(x)) &= (c(x), 0) \cdot B_{f,g}^{-1} = \\ &= (c(x), 0) \cdot \frac{1}{q} \begin{bmatrix} -F(x) & f(x) \\ -G(x) & g(x) \end{bmatrix} = \\ &= \left(-\frac{1}{q} F(x) \star c(x), \frac{1}{q} f(x) \cdot c(x) \right) \end{aligned} \tag{6.30}$$

Si calcola $(z_0(x), z_1(x))$ mediante il Fast Fourier Sampling con input l'albero e $(t_0(x), t_1(x))$, allora

$$(s_0(x), s_1(x)) = ((t_0(x), t_1(x)) - (z_0(x), z_1(x)))B_{f,g} \tag{6.31}$$

Visto che $(z_0(x), z_1(x)) \in \mathbb{Z}[x]/(x^N + 1)$, $(z_0(x), z_1(x))B_{f,g} \in \Lambda^{\perp}$ e dunque

$$(s_0(x), s_1(x)) \cdot A = s_0(x) + s_1(x) \star h(x) = c(x) \tag{6.32}$$

cioè $(s_0(x), s_1(x))$ è soluzione della stessa equazione soddisfatta da $(t_0(x), t_1(x))B_{f,g}$. Se $(z_0(x), z_1(x)B_{f,g}$ è un elemento abbastanza vicino a $(t_0(x), t_1(x))B_{f,g}$, la norma di $(s_0(x), s_1(x))$ è piccola, come si desidera.

Anche la firma può utilizzare come format la rappresentazione FFT.

6.5.1 Fast Fourier Sampling

Algorithm 10 ffSampling$_N(t, T)$

Input: $\underline{t} = (\underline{t}_0, \underline{t}_1) \in FFT((\mathbb{Q}[x]/(x^N + 1))^2)$, T albero FALCON
Output: $\underline{z} = (\underline{z}_0, \underline{z}_1) \in FFT(R^2)$

if $N = 1$ **then**
  $\sigma' \leftarrow$ T.value
  $z_0 \leftarrow$ Sampler$\mathbb{Z}(t_0, \sigma')$
  $z_1 \leftarrow$ Sampler$\mathbb{Z}(t_1, \sigma')$
  return $\underline{z} = (z_0, z_1)$
else
  $(\underline{\ell}, \underline{T}_0, \underline{T}_1) \leftarrow$ (T.value, T.leftchild, T.rightchild)
  $\underline{\mathbf{t_1}} \leftarrow$ splitFFT$(\underline{t}_1)$
  $\underline{\mathbf{z_1}} \leftarrow$ ffSampling$_{N/2}(\underline{\mathbf{t_1}}, \underline{T}_1)$
  $\underline{z}_1 \leftarrow$ mergeFFT$(\underline{\mathbf{z_1}})$
  $\underline{t}'_0 \leftarrow \underline{t}_0 + (\underline{t}_1 - \underline{z}_1) \cdot \underline{\ell}$
  $\underline{\mathbf{t_0}} \leftarrow$ splitFFT$(\underline{t}'_0)$
  $\underline{\mathbf{z_0}} \leftarrow$ ffSampling$_{N/2}(\underline{\mathbf{t_0}}, \underline{T}_0)$
  $\underline{z}_0 \leftarrow$ mergeFFT$(\underline{\mathbf{z_0}})$
  return $\underline{z} = (\underline{z}_0, \underline{z}_1)$
end if

Nell'algoritmo 10, Sampler$\mathbb{Z}$ con input μ, σ' denota un algoritmo di campionamento dalla distribuzione $D_{\mathbb{Z},\sigma',\mu}$. Esso può essere l'algoritmo indicato nel GPV framework o altri algoritmi.

Si pubblica il messaggio $\underline{m}$ corredato della firma $\underline{r}$, e la seconda metà delle entrate di $(\underline{t} - \underline{z})B_{f,g}$.

6.5.2 Verifica

- Si calcola $c(x) = H(\underline{r}||\underline{m})$.
- Si calcola $s_0(x) = c(x) - s_1(x) \star h(x) \mod q$
- Si controlla se $\|(s_0(x), s_1(x))\|^2 \leq \lfloor \beta^2 \rfloor$, se così non è la firma viene rifiutata in quanto falsa.

6.5.3 Esempio

Si illustra un esempio di quanto visto con parametri piccoli, affinché sia di facile comprensione. Si sottolinea che con parametri così piccoli non c'è la sicurezza garantita dalle scelte suggerite dagli autori.

Sia $N = 2$, $q = 17$.

Il primo passo è la generazione delle chiavi private $f(x)$ e $g(x)$: si tratta di due polinomi in $\mathbb{Z}[x]/(x^N+1)$ il cui vettore dei coefficienti viene campionato da una distribuzione Gaussiana discreta su $\mathbb{Z}^N$ centrata in 0 e con deviazione standard $\frac{1.17\sqrt{q}}{2}$, affinché il valor medio della norma di $(g(x), -f(x))$ sia $1.17\sqrt{q}$. Questi due polinomi devono essere invertibili modulo 17 e tali che la risultante base B abbia norma $\|B\|_{GS}$ minore o uguale a $1.17\sqrt{q}$, per minimizzare la norma delle firme. Non viene soddisfatta l'ultima condizione a causa della scelta semplice dei parametri, ma comunque ciò non inficia l'esempio, perché $1.17\sqrt{q} \approx 4.824$ è poco più piccolo di $\|\tilde{\underline{b}}_{N+1}\|$.

Una scelta ottimale di $f(x)$ e $g(x)$ è data da

$$f(x) = x + 1 \qquad g(x) = x - 2 \tag{6.33}$$

Infatti

$$\|(g(x), -f(x))\|_2 = \|(1, 1, 1, -2)\|_2 = \sqrt{7} \approx 2.646$$

$$\left\| \left(\frac{qf^\star(x)}{f \star (x) f^\star(x) + g(x) \star g^\star(x)}, \frac{qg^\star(x)}{f(x) \star f^\star(x) + g(x) \star g^\star(x)} \right) \right\|_2 =$$
$$= \left\| \left(\frac{17(-x+1)}{7}, \frac{17(-x-2)}{7} \right) \right\|_2 = \frac{17}{7} \|(-1, 1-1, -2)\|_2 \approx 6.425$$

Perciò $\|B_{f,g}\| = \max\{2.646, 6.425\}$=6.425.

La chiave pubblica si determina a partire dal polinomio

$$h(x) = f(x)^{-1} g(x) \mod 17 = 9(1-x)(x-2) = 9(3x-1) = 10x - 9 \mod 17$$

Si risolve l'equazione NTRU associata a $f(x), g(x)$.

$$f'(x) = N(f(x)) = (x+1)(-x+1) = 2 \qquad g'(x) = N(g(x)) = (x-2)(-x-2) = 5$$

$$\gcd(2, 5) = 1 \Longrightarrow 2u - 5v = 1$$

Una possibile soluzione è $(u, v) = (-2, -1)$.

$$F'(x) = qv = -17 \quad G'(x) = qu = -34$$

Infine

$$F(x) = F'(x^2) g(-x) = -17(-x-2) = 17(x+2)$$
$$G(x) = G'(x^2) f(-x) = -34(-x+1) = 34(x-1)$$

Resta solo da ridurre i due polinomi:

$$k(x) = \left\lfloor \frac{f^\star(x) \star F(x) + g^\star(x) \star G(x)}{f(x) \star f^\star(x) + g(x) \star g^\star(x)} \right\rceil = -7x + 21$$

$$\begin{aligned} F(x) &:= F(x) - k(x) \star f(x) = \\ &= 17(x+2) - (-7x+21) \star (x+1) = 17x + 34 - 14x - 28 = \\ &= 3x + 6 \end{aligned}$$

$$\begin{aligned} G(x) &:= G(x) - g(x) \star f(x) = \\ &= 34(x-1) - (-7x+21) \star (x-2) = 34x - 34 - 35x + 35 = \\ &= -x + 1 \end{aligned}$$

Si calcola ora l'albero LDL* della matrice $B_{f,g}B^\star_{f,g}$.

$$G = B_{f,g}B^\star_{f,g} = \begin{bmatrix} g(x) & -f(x) \\ G(x) & -F(x) \end{bmatrix} \begin{bmatrix} g^\star(x) & G^\star(x) \\ -f^\star(x) & -F^\star(x) \end{bmatrix} = \begin{bmatrix} 7 & 2x+6 \\ -2x+6 & 47 \end{bmatrix}$$

$$\begin{aligned} D_1(x) &= G_{1,1}(x) = 7 \\ L_{2,1}(x) &= \frac{G_{2,1}(x)}{D_1(x)} = \frac{1}{7}(-2x+6) = \frac{2}{7}(-x+3) \\ D_2(x) &= G_{2,2}(x) - D_1(x) \star L_{2,1}(x) \star L^\star_{2,1}(x) = \frac{289}{7} \end{aligned}$$

Si ottiene l'albero in 6.1.

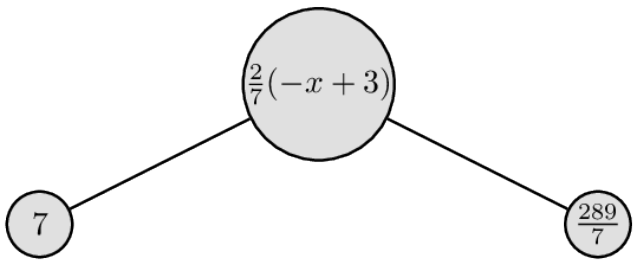


Figura 6.1: Albero LDL*

Per renderlo un albero FALCON si sceglie un valore $\sigma \geq \|B\|_{GS}\eta_\epsilon(\mathbb{Z})$, ad esempio 7, e i valori delle foglie dell'albero vengono sostituiti con σ diviso per la radice quadrata del valore della foglia. Il risultato di questa operazione è l'albero 6.2

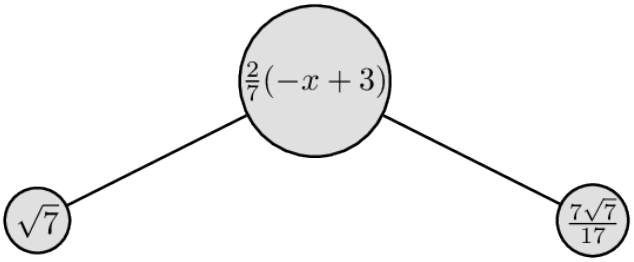


Figura 6.2: Albero FALCON

Si suppone di voler firmare il message digest $c(x) = 2$. Dapprima si determina una controimmagine di $c(x)$:

$$\begin{aligned}\underline{t} &= (t_0(x), t_1(x)) = (c(x), 0) \cdot B_{f,g}^{-1} = \\ &= \frac{1}{17}(-c(x) \star F(X), c(x) \star f(x)) = \\ &= \left(-\frac{6}{17}(x+2), \frac{2}{17}(x+1)\right)\end{aligned}$$

Si può ora applicare l'algoritmo di campionamento per trovare $\underline{z} = (z_0(x), z_1(x))$. Per maggior chiarezza, non si utilizza la rappresentazione FFT.

$$l(x) = \frac{2}{7}(-x+3) \quad T_0 = \sqrt{7} \quad T_1 = \frac{7\sqrt{7}}{17}$$

$$\underline{t}_1 = \text{split}(t_1(x)) = \left(\frac{2}{17}, \frac{2}{17}\right)$$

$$\underline{z}_1 = \text{Sampling}(\underline{t}_1, T_1) = (0, 1)$$

$$z_1(x) = \text{merge}(\underline{z}_1) = x$$

$$\begin{aligned}t_0'(x) &= t_0(x) + (t_1(x) - z_1(x)) \star l(x) = \\ &= \frac{1}{119}(-136x - 102)\end{aligned}$$

$$\underline{t}_0 = \text{split}(t_0'(x)) = \left(-\frac{102}{119}, -\frac{136}{119}\right)$$

$$\underline{z}_0 = \text{Sampling}(\underline{t}_0, T_0) = (-1, -1)$$

$$z_0(x) = \text{merge}(\underline{z}_0) = -x - 1$$

Perciò il risultato finale è

$$\underline{z} = (z_0(x), z_1(x)) = (x, -x - 1)$$

$$\begin{aligned}\underline{s} &= (s_0(x), s_1(x)) = (\underline{t} - \underline{z})B_{f,g} = \\ &= \left(\frac{11}{17}x + \frac{5}{17}, -\frac{15}{17} + \frac{2}{17}\right)\begin{bmatrix} x-2 & -(x+1) \\ -x+1 & -3(x+2)\end{bmatrix} = \\ &= (-2x-2, 4x-3)\end{aligned}$$

Alice correda il messaggio della firma $s_1(x)$ e invia il tutto a Bob, insieme al bound sulla norma 6.

Alice può verificare che effettivamente

$$\begin{aligned}s_0(x) + s_1(x) \star h(x) &= -2x - 2 + (4x-3) \star (10x-9) = \\ &= -2x - 2 + 2x + 4 = 2 = c(x) \mod 17\end{aligned}$$

Bob riceve la coppia messaggio e firma, e può calcolare il message digest, perché la funzione di Hash è pubblica, dunque ottiene $c(x) = 2$. Ora Bob può calcolare la prima metà della firma:

$$s_0(x) = c(x) - s_1(x) \star h(x) \mod 17 = -2x - 2 \mod 17$$

Infine Bob controlla che la norma della firma sia inferiore al limite dato da Alice, cioè che la firma sia vera:

$$\|s_0(x), s_1(x)\|_2 = \|(-2, -2, 4, -3)\|_2 = \sqrt{33} \approx 5.745 < 6$$

Bibliografia

[1] M. Ajtai *Generating hard istances of the short basis problem* In ICALP, pages 1-9, 1999.

[2] W. Banaszczyk. *New bounds in some transference theorems in the geometry of numbers* Mathematische Annalen, 296(4):625-635, 1993.

[3] F. Benevenuto *Problemi Inversi Appunti della seconda parte del corso* Università di Genova, 2021

[4] D.J. Bernestein, J. Buchmann, E. Dahmen *Post-Quantum Cryptography* Springer- Verlag Berlin Heidelberg, 2009

[5] M. Bertuzzo. *Lattice-based Criptography* Corso di Laurea Magistrale in Matematica, Università di Genova (2023).

[6] D.A. Bini. *La trasformata discreta di Fourier e la FFT* Università di Pisa, 7 febbraio 2020

[7] D. Boneh, Ö. Dagdelen, M. Fischlin, A. Lehmann, C. Schaffner, M. Zhandry. *Random oracles in a quantum world* In Dong Hon Lee e Xiaoyun Wang, ASIACRYPT 2011, volume 7073 di LNCS, Corea del Sud, 2011, Springer, Heidelberg.

[8] J. S. Cohen *Computer Algebra and Symbolic Computation*, A K Peters, Ltd, 2003.

[9] J. W. Cooley, J. W. Tukey *An Algorithm for the Machine Calculation of Complex Fourier Series* Mathematics of Computation, 1965

[10] L. Ducas, V. Lyubashevsky, T. Prest *Efficient identity-based encryption over NTRU lattices.* In Palash Sarkar and Tetsu Iwata, editors, ASIACYPT 2014, Part II, Kaoshiung, Taiwan, December 7-11, 2014. Springer, Heidelberg, Germany.

[11] L. Ducas, P.Q. Nguyen *Faster gaussian lattice sampling using lazy floating-point arithmetic.* ASIACRYPT '12, Pagine 433-450, Berlino, Heilderberg, Springer-Verlag, 2012.

[12] L. Ducas, T. Prest *Fast Fourier Orthogonalization* ISAAC'16: Proceedings of the 2016 ACM International Symposium on Symbolic and Algebraic Computation Pagine 191-198

[13] P.A. Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Prest, T. Ricosset, G. Seiler, W. Whyte, Z. Zhang *FALCON: Fast-Fourier Lattice-based Compact Signatures over NTRU* Specification v1.2-01/10/2020

[14] S. D. Galbraith *Mathematics of Public Key Criptography. Version 2.0* Cambridge University Press, 2018.

[15] C. Gentry, C. Peikert, V. Vaikuntanathan. *How to Use a Short Basis: Trapdoors for Hard Lattices and New Cryptographic Constructions* STOC 2008.

[16] J. Hoffstein, N. Howgrave-Graham, J.Pipher, J.H. Silverman, W. Whyte *NTRUSign: Digital Signatures Using the NTRU Lattice* Topics in Cryptology — CT-RSA 2003. LNCS. Vol. 2612. Springer. pp. 122–140.

[17] J. Hoffstein, N. Howgrave-Graham, J.Pipher, J.H. Silverman, W. Whyte *NTRUSign: Digital Signatures Using the NTRU Lattice* Preliminary Draft 2, April 2,2002.

[18] J. Hoffstein, J. Pipher, J.H. Silverman *NTRU: A New High Speed Public Key Cryptosystem* Preprint; presented at the rump session of Crypto 1996

[19] J. Hoffstein, J. Pipher, J.H. Silverman *NTRU: A Ring-Based Public Key Cryptosystem* 1998

[20] J. Hoffstein, J. Pipher, J. H. Silverman. *An Introduction to Mathematical Cryptography* Undergraduate Texts in Mathematics, 2nd Edition Springer-Verlag, 2014.

[21] R. Lidl, H. Niederreiter *Finite fields* Encyclopedia of Mathematics and its Applications, 2nd edition Cambridge University Press, 1996

[22] D. Micciancio, S. Goldwasser *Complexity of Lattice Problems: a cryptographic perspective* The Kluwer International Series in Engineering and Computer Science, 2002

[23] J.Neukirch *Algebraic Number Theory* Tradotto dal tedesco da N. Schappacher, Springer-Verlag Berlin Heidelberg 1999

[24] P. Q. Nguyen, O. Regev *Learning a Parallelepiped: Cryptanalysis of GGH and NTRU Signatures* In: Vaudenay, S. (eds) Advances in Cryptology - EUROCRYPT 2006.

[25] D. Stehlé, R. Steinfeld *Making NTRU as Secure as Worst-Case Problems over Ideal Lattices* In Kenneth G. Paterson, editor, EUROCRYPT 2011, volume 6632 of LNCS, pages 27-47, Tallinn, Estonia, May 15-19, 2011. Springer, Heidelberg, Germany.

[26] D. R. Stinson, M. Paterson. *Cryptography Theory and Practice. Fourth Edition*, Chapman & Hall/CRC, 2019.